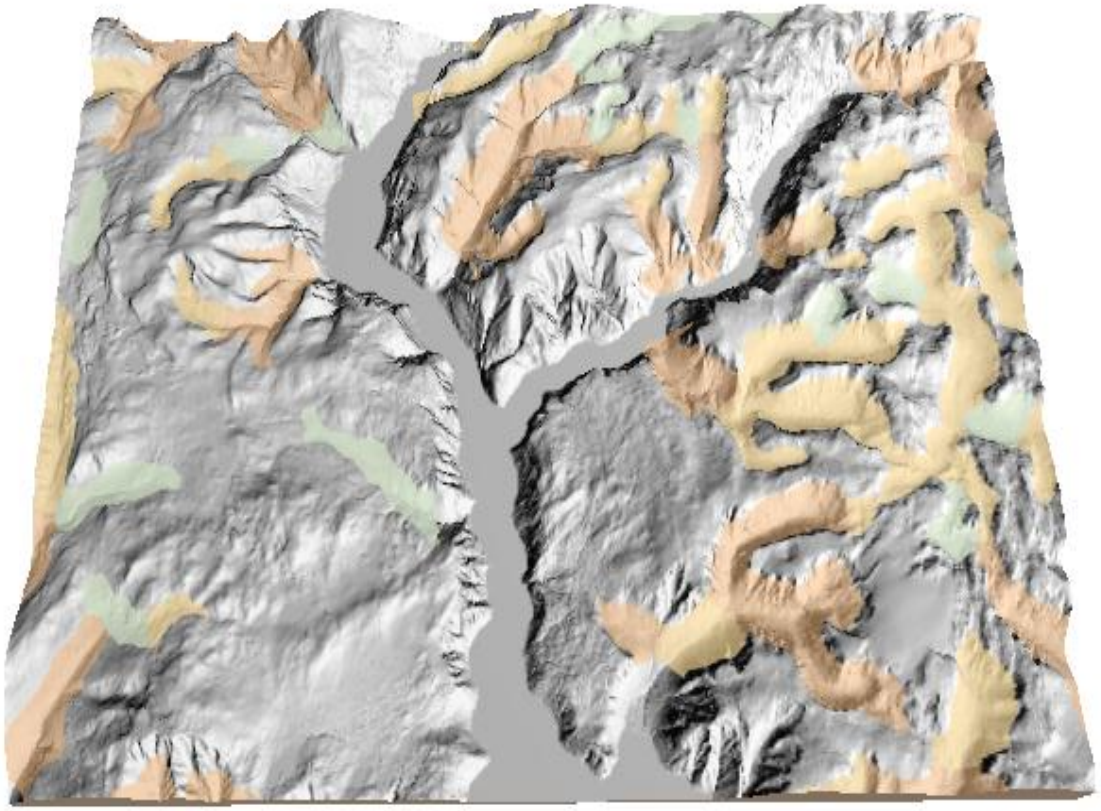
Master Thesis, Department of Geosciences

Automatic detection of valley forms

Nagy Robert



UNIVERSITY OF OSLO

FACULTY OF MATHEMATICS AND NATURAL SCIENCES

Automatic detection of valley forms

Nagy Robert

Master Thesis in Geosciences

Discipline: Phisical Geography, Hydrology and Geomatics

Department of Geosciences

Faculty of Mathematics and Natural Sciences

University of Oslo

03.07.2015

ABSTRACT

Landform detection and analysis has been an important topic since the advent of digital elevation models. In Scandinavia valleys were mostly formed by erosion of water (fluvial) or ice (glacial), resulting in v-shaped valleys and u-shaped valleys. Valleys and their distribution is an important indicator for former ice-sheet configuration, temperature regimes and landscape development.

In this study a method is proposed to detect valleys and their form. The aforementioned method which can be used at different scale, can handle directional changes in the valley path. The algorithm uses, mathematical morphology, image skeletons, kNN classification and Multivariate Gaussian classification to tackle the complex problem of valley detection using the topographic parameters slope and curvature.

The method is then assessed on the 10 meter and 50 meter elevation grid from the Norwegian Mapping Authority, ASTER GDEM, and SRTM. Overall results show that the 10 meter elevation grid yields the best results with 80% of the v-valley samples, 87% of the u samples, and 100% of the filled up valleys being correctly classified. ASTER GDEM produces poor results because of the high amount of noise.

Front cover page: shaded relief of area near Nærøyfjorden and Bleia Storebotnen

Acknowledgments

First and foremost I would like to thank Professor Bernd Etzelmuller and Professor Karianne Staalesen Lilleøren for the outstanding guidance I have received. Professor Bernd, thank you for the smiles and the kind words that you have shared, they were both helpful and inspirational. I have never seen a professor be so student oriented. I cannot show enough appreciation towards your hard work.

I would like to thank Horvath-Tholdy Eva, who has been not just a mother but a friend, and who has been helping me become what I am. Köszönöm édesanyam hogy támogattal engemet! Special thanks go also to Nagy Francisc, my father, who has helped me financially through my master studies, helping me to achieve my dreams. Mutlumesc tată că mai ajutat sa imi realizez visul! Fără tine nu as fi reușit să fac asta!

A big shout out to my best friends Molnár István, Herghelegiu Andrei, Molnár Jenő, Bokor Csaba. Yo dudes! I made it! Thanks for the laughs, the jokes, the smiles, the Counter Strike matches, and the kind words!

Special thanks to the friends from room 214 (Yes Amund! Room 214 not room 219!) We had so many nice pranks that we made on each other, I will never forget them. Ida, thanks for the delicious cakes, Celine, Mathieu, Oda and Alex thanks for the talks, and laughs. Di Wang, thank you for the amazing trip to China! I will never forget the effort that you made for me! It was a great and educational trip that we made together. Also thank you for the countless times you helped me in Matlab!

A special thanks to the people who were hanging around in the kitchen of the ZEB building. May our cake/waffle Fridays live on among the generations. Thank you Gauti for the meaningless and useless but yet therapeutical philosophical conversations.

And a special thank you for Magdalini for helping me with good tips in life, and the assistance in this thesis. You are a good friend Magda.

Table of contents

List of figures

List of Tables

1. Introduction

Valleys on the Earth's surface are morphological expressions of tectonics (upheave) and linear erosion. For Scandinavia the agents of erosion were water (fluvial), ice (glacial) or a combination of both. Rivers alone normally produce v-formed valleys while erosive ice streams fill such valleys and transform them into u-shaped valley forms. For a formerly glaciated landscape like Norway one would expect a dominance of u-shaped valleys, which is certainly the case for most of the Norwegian coastal areas. However, Norway presents an abundance of v-shaped valleys, indicating fluvial erosion after the last glaciation or preservation of older valley systems of different reasons. Thus, valley forms and their distribution are an important indicator for former ice-sheet configuration, temperature regimes (cold based non-erosive ice) and landscape development (Etzelmuller, 2015).

True valleys are linear depressions on the surface of Earth, which invariably are longer than they are wide with floors sloping downwards. Valleys are very common landforms which tend to be overlooked by geomorphologists (Huggett, 2003). Usually a product of corrosion, abrasion, pot-holing, cavitation, weathering(Goudie, 2004) they develop networks and they have different drainage patterns. The types of valleys include: gully, draw, defile, ravine, gulch, hollow, run arroyo, gorge, canyon, dell, glen, dale and vale (Huggett, 2003).

Visually, it is not hard to identify a valley on a digital elevation model, but automating such a multi-scale, multi-directional landform can be a challenge. Such a method, that can handle both directionality and scale is mathematical morphology invented by Serra (1986) and was first applied to a digital elevation model by Rodriguez et al. (2002). Using a structuring element of a given size, one could easily identify both valleys and mountain tops in a simple yet ingenious way.

Identifying valleys is vital as it establishes an understanding of how the topography was modeled, and also what kind of powers have helped to shape the landscape in the current form.

1.1 Aim

This thesis seeks to develop a methodology to automatically detect major valley forms based on DEMs, applicable for large areas. This allows for analyzing quantitatively the distributional pattern of valley forms, and hence contributes with important information about the development of the topography along a passive margin like the one identified in western Scandinavia. The thesis is restricted to the development of the methodology, while the analyses of patterns are a next important step, but not addressed here(Etzelmuller, 2015).

Old concepts of the evolution of Scandinavia have been challenged, and the identification of valleys would be important to understand for example off-shore geology in the North – Sea . The aim of this study is to find an automatic method to identify and classify valleys in Norway. The methodology used is expected to be able to cope with scalability and directionality of the valleys. It should also be able to handle local variations within valleys, for example flat areas as valley bottoms can be flat in some cases.. Mainly three types of valleys are aimed to be identified, and the forth identified objects are fjords, which are the product of a reclassification. The main three types of valleys which will be identified are u – shaped valleys, v – shaped valleys, and filled up valleys.

1.2 Structure of the thesis

The thesis is structured into 6 chapters. Chapter 1 starts with the introduction of the thesis, chapter 2 describes the basic theoretical concepts behind the hat transform(White-Top-Hat/Black-Top-Hat), image skeletons, connected component labeling, kNN classification and the Multivariate Gaussian classification, whereas chapter 3 describes the methodology used. Chapter 4 describes the result and discussions of the valley detection algorithm, chapter 5 the discussions, and chapter 6 contains the conclusions and outlook . The seventh chapter is reserved for the reference list. The thesis also includes appendices: appendix A,B is the python code used to identify the valleys, appendix I,II,II represents the data used to train the multivariate Gaussian classifier for the 10 meter resolution elevation model from The Norwegian Mapping authority, Appendix IV.A,IV.B,IV.C, data which was used to train the Gaussian classifier for aster GDEM, Appendix V, VI, data which were used to assess the accuracy of the classification.

2. Theoretical framework

2.1 Valleys

Detecting landforms on earth's surface has been an important topic since the appearance of digital elevation models. A general classification of landform elements has been successfully made using fuzzy logic and logistic rules (MacMillan et al., 2000, Burrough et al., 2000, Schmidt and Hewitt, 2004), mean curvature watersheds (Romstad and Etzelmüller, 2012) ,OBIA (Object Based Image Analysis)(Drăguţ and Blaschke, 2006, Gerçek et al., 2011) , the latter classifying mainly into the 9 basic landform elements defined by Dikau (1989).

Valleys can well be defined by basic landform elements presented by Dikau, and also further completed by (Schmidt and Hewitt, 2004) displayed in figure 1.0, in theory, but in reality the problem is more complex due to the variation of topography. Using he combination of tangential and profile curvature, v- valleys would be mostly composed of shoulder slopes, and U – Valleys would mostly be composed of foot slopes. In reality, the problem demonstrates further intricacies, and valley sides are characterized by high variations.

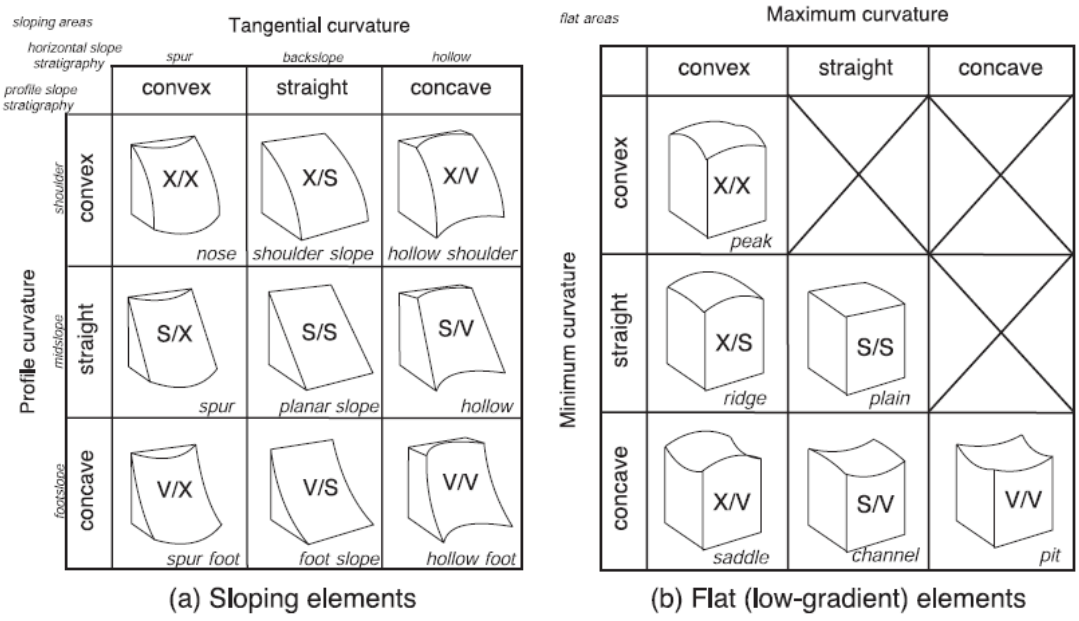


Figure 1.0 Theoretical terrain units classified by tangential curvature and profile curvature after Schmidt and Hewitt (2004)

2.1.1 V – Shaped Valleys

Carved and created by fluvial processes, v – shaped valleys are erosional landforms, with concave river profiles from source to mouth(Huggett, 2003). These valleys have usually side slopes under 30-33 degrees, the valley floors are also steep, around 30 degrees, and dominantly the profile curvature is usually from straight to convex (Etzelmuller, 2014). Displayed in figure 1.1 is the 3D view of a v – shaped valley, and as a characteristic, the floor is narrow, and the sides as the name suggests V shaped. A better representation of this can be examined in figure 1.2, which represents a cross section of the valley.

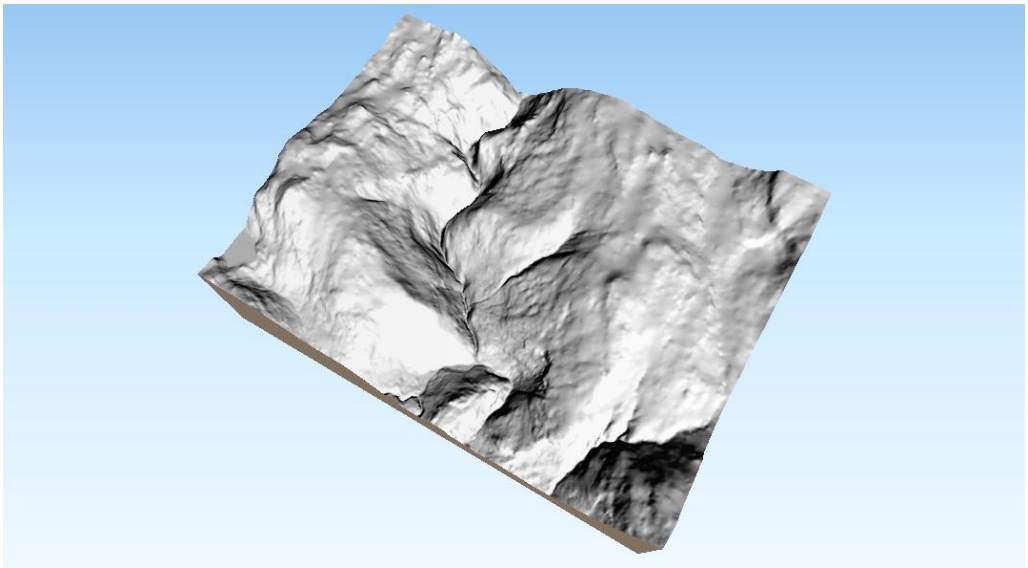


Figure 1.1 3D view of a V – Valley. The valley bottom is narrow and the valley sides are convex

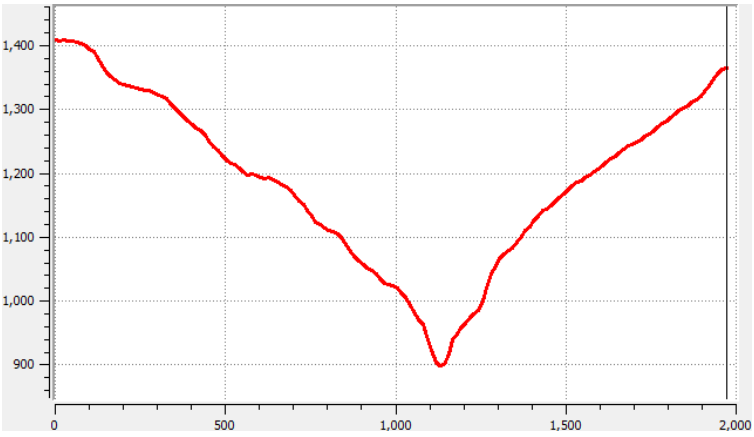


Figure 1.2 V shaped valley. High convexity can be seen on the sides of the cross section.

2.1.2 U – Valley and Fjords

U – Shaped valleys as well as fjords are the result of glacial erosion. A glaciated valley floor lies above the sea level, whereas the floor of the fjord lies under the sea level. Hanging valleys are also a product of glacial erosion, usually a little smaller (Huggett, 2003). Due to the high erosion of the snow, u – shaped valleys usually have steeper sides, with steepness above 35 degrees and the dominant profile curvature within is from straight to concave(Etzelmuller, 2014). Figure 1.3 shows the cross section of a u shaped valley (left) and the cross section of a fjord (right). A better view of how steep the walls of u valleys are, is visible in figure 1.4, the valley floor is signifficantly wider than in the case of v - valleys

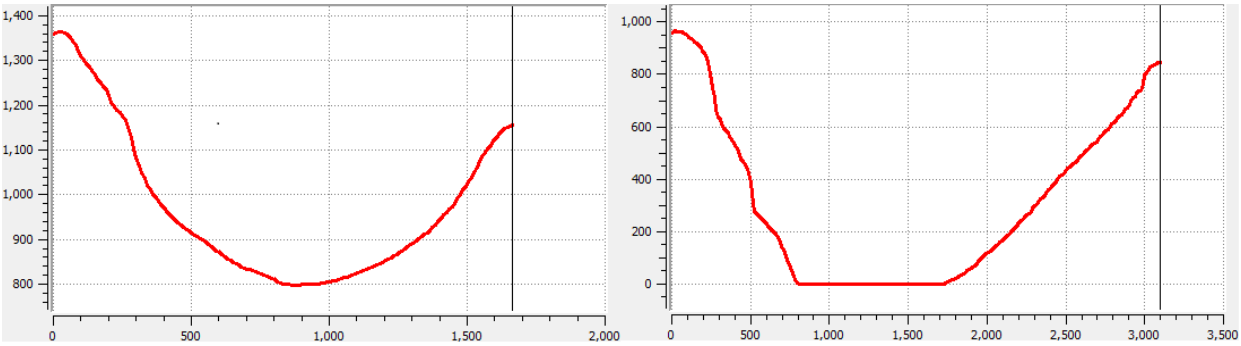


Figure 1.3 On the left side a cross section of a u – shaped valley is displayed, on the right a cross section of a fjord. It can be seen that the bottom of the fjord is at sea level

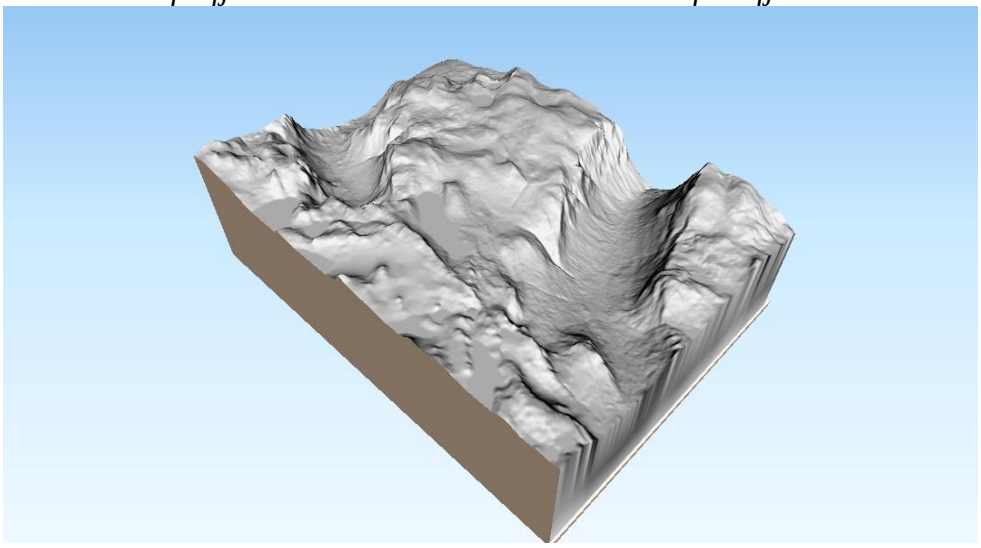


Figure 1.4 3D view of u shaped valley. The valley bottom is wider, and the valley sides are steeper

2.1.3 Filled up Valleys

In essence, filled up valleys exhibit the same characteristics as u-shaped valleys. They are usually filled up with debris and/or water, the valley walls are also concave, and the elevation of the valley bottom is higher than sea level, which is the main difference to fjords. A 3D view of such a valley is visible in figure 1.5. In this case the valley bottom is filled up with water. The cross section of this is observable in figure 1.6

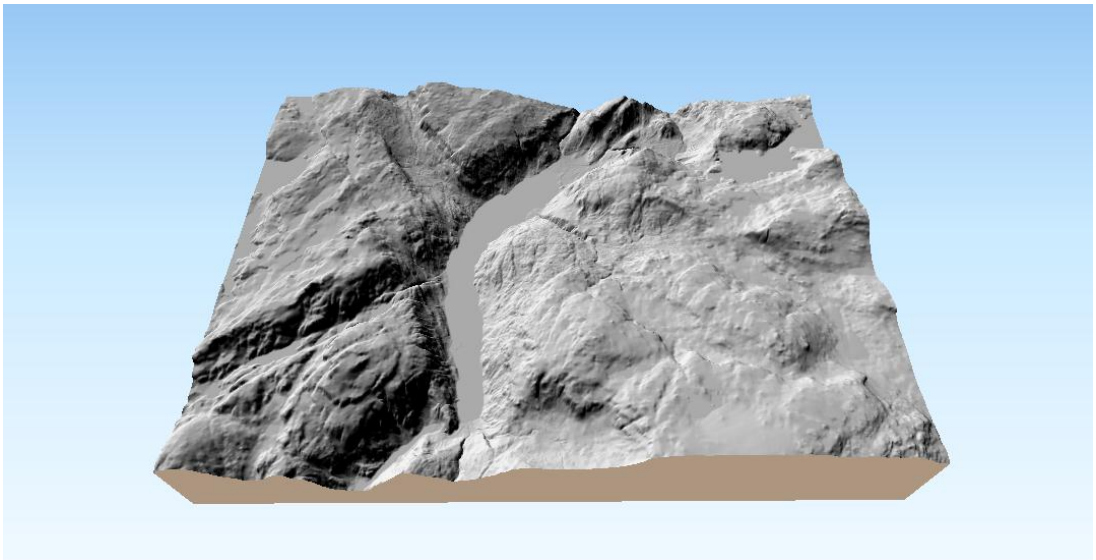


Figure 1.5 3D view of filled up valley. The valley sides are concave and the bottom is filled with water

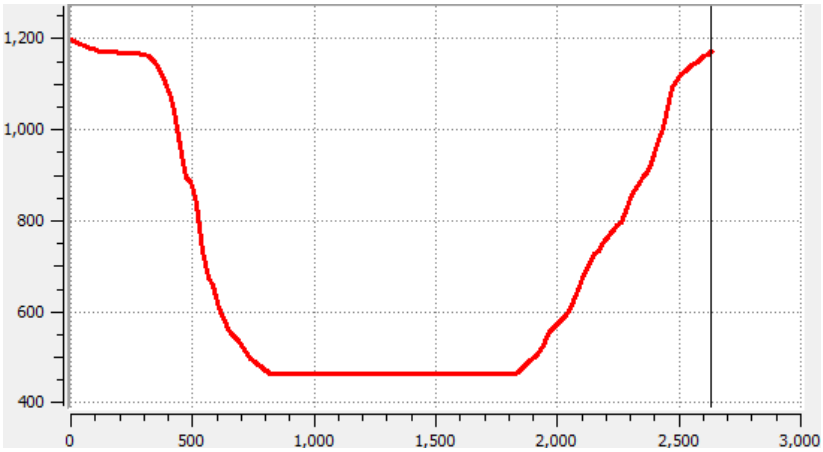


Figure 1.6 Filled up Valley. Similarly to U shaped valleys it has high slopes, but the elevation of debris/water is not at sea level.

2.2 Topographic parameters

The importance of the topographic parameters is recognized largely by engineers and geomorphologists. The aforementioned parameters are widely used in the study of land surfaces and the topographic effects on hydrologic and sedimentary processes(Zevenbergen and Thorne, 1987). A topographic parameter is the result of an analysis of the geometrical proprieties of the land surface. These parameters are: slope, aspect, curvatures as well as other values which can result from them. These can be derived in a mathematical way, for instance it is common to use concepts from differential geometry, but can be computed locally using a kernel (3x3 window)(Hengl and Reuter, 2009). Slope influences flow rates of water and sediment. Aspect defines the slope direction therefore the flow direction. Profile curvature represents the rate of change of slope, which determines the flow acceleration/deceleration. Plan curvature influences the divergence or convergence of flows.(Zevenbergen and Thorne, 1987)

2.2.1 Slope

Slope is a first order derivative of the terrain. For the estimation of this parameter the gradient needs to be calculated. The gradient is nothing else than the vector field pointing in the direction with maximal variation in the values of the scalar field.(Hengl and Reuter, 2009)

$$\nabla \bar{Z} = \left(\frac{\delta z}{\delta x}, \frac{\delta z}{\delta y}\right) \quad (1)$$

Slope and aspect are two meaningful geometric parameters derived from the gradient. Slope gradient reflects the maximal rate of change of elevation values and is defined as follows:

$$SLOPE = \arctan(|\nabla \bar{Z}|) \quad (2)$$

Using Evans method one can approximate the first order derivatives in x and y direction in the following way:

$$\frac{\delta z}{\delta x} = \frac{z_3 + z_6 + z_9 - z_1 - z_4 - z_7}{6 * \Delta s} \quad (3)$$

$$\frac{\delta z}{\delta x} = \frac{z_3 + z_6 + z_9 - z_1 - z_4 - z_7}{6 * \Delta s} \quad (4)$$

Applying the arctan function to the first order derivatives the formula of slope becomes:

$$SLOPE = \arctan(\sqrt{\left(\left(\frac{\delta z}{\delta x}\right)^2 + \left(\frac{\delta z}{\delta y}\right)^2\right)} \quad (5)$$

A slope map can be calculated in radians, degrees and percentage. The values are always positive and always within the $\left[0, \frac{\pi}{2}\right]$ interval. Figure 2.1 depicts a slope map in radians, where areas with low slopes are dark and areas with high slopes are bright.

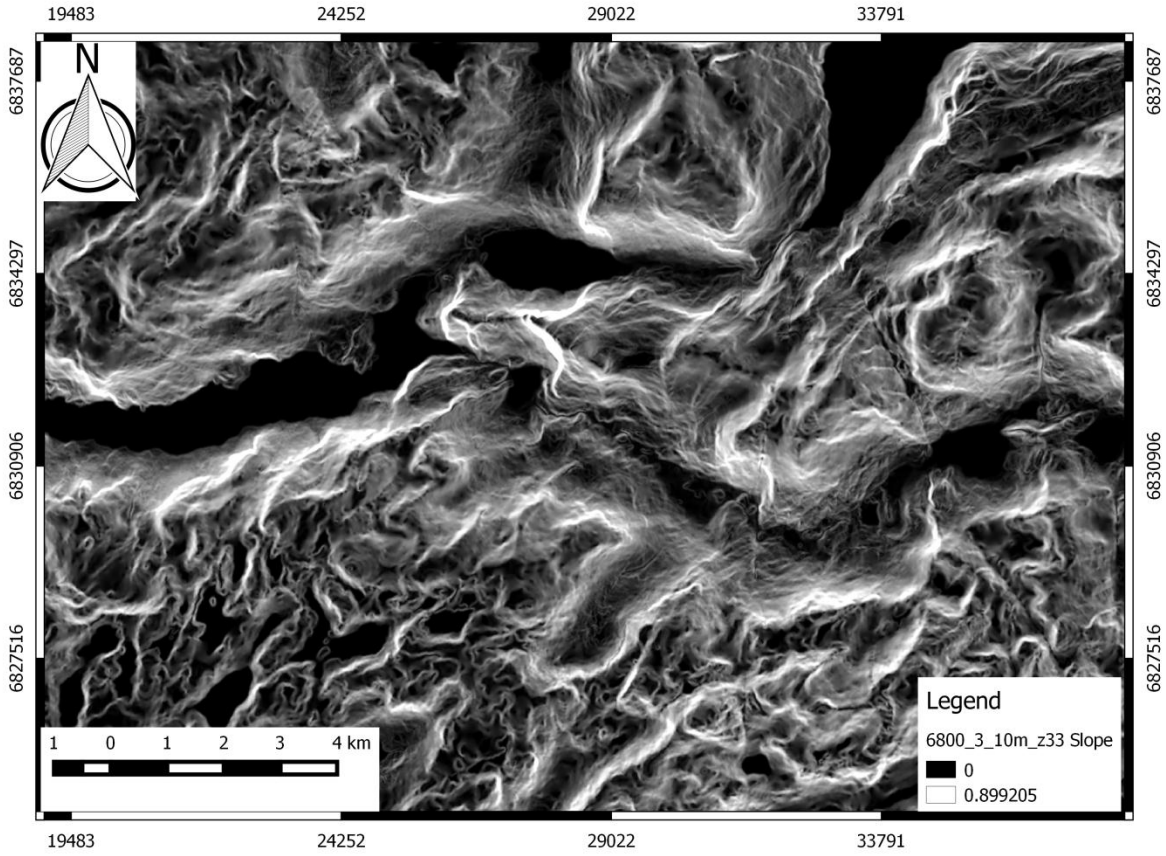


Figure 2.1 Slope map derived from a digital elevation model. Areas with high slopes are dark, and areas with high slope are brighter in color.

2.2.2 Curvature

Curvature is a local morphometric variable, which can be calculated using the second order derivative. The two parameters which describe curvature are profile curvature and tangential curvature both of them expressing the concavity or convexity of a surface(Hengl and Reuter, 2009). Shown in figure 2.2, the profile curvature is that of normal section aa', and the tangential curvature is that of the normal section bb'(Hengl and Reuter, 2009)

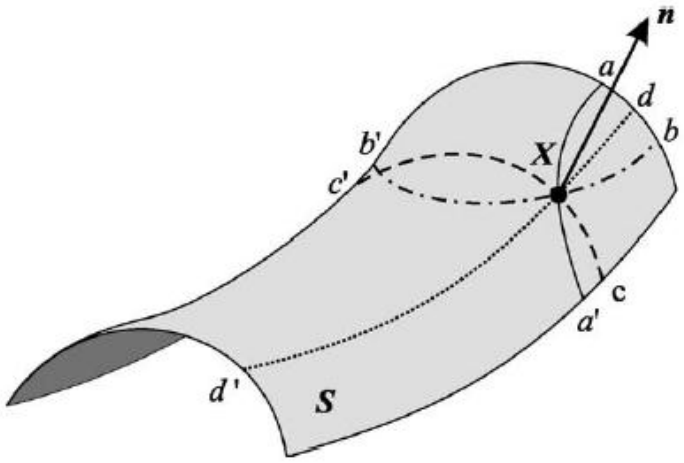


Figure 2.2 The four directions naturally marked on surface S.n – the normal vector to S at point X; aa' – the gradient line, bb' - the contour line, dd', cc' – the main normal sections. After Hengl and Reuter (2009).

The formulas to determine the curvatures are:

$$r = \frac{\partial^2 z}{\partial x^2,} \quad (6) \qquad s = \frac{\partial^2 z}{\partial x \partial y,} \quad (7) \qquad t = \frac{\partial^2 z}{\partial y^2} \quad (8)$$

Using Evans (1972) method, the secondary derivatives will receive the following values:

$$r = \frac{z_1 + z_3 + z_4 + z_6 + z_7 + z_9 - 2 * (z_2 + z_5 + z_8)}{3 * \Delta s^2} \quad (9)$$

$$s = \frac{z_3 + z_7 - z_1 - z_9}{4 * \Delta s^2} \quad (10)$$

$$t = \frac{z_1 + z_2 + z_3 + z_7 + z_8 + z_9 - 2 * (z_4 + z_5 + z_6)}{3 * \Delta s^2} \quad (11)$$

Using the expressions from above, the profile curvature can be expressed with the formulas (Krcho and Haverlik, 1973, Young and Evans, 1978), where the slope is higher than zero, and is field specific(Hengl and Reuter, 2009):

$$PROFC = -\frac{p^2 * r + 2 * p * q * r * s + q^2 * t}{(p^2 + q^2) * \sqrt{(1 + p^2 + q^2)^3}} \quad (12)$$

Similarly not defined in special points and also field specific, the tangential curvature can be derived using (Krcho, 1983, Shary, 1991, Mitasova et al., 1996):

$$TANGC = -\frac{q^2 * r - 2 * p * q * r * s + p^2 * t}{(p^2 + q^2) * \sqrt{1 + p^2 + q^2}} \quad (13)$$

The average curvature of a section can be calculated using the profile curvature and the tangential curvature, by averaging them (Hengl and Reuter, 2009)

$$MEANC = \frac{PROFC + TANGC}{2} \quad (14)$$

Displayed in figure 2.3 is the result of averaging the profile curvature and the tangential curvature. The result is the mean curvature, where the values can be both negative and positive .

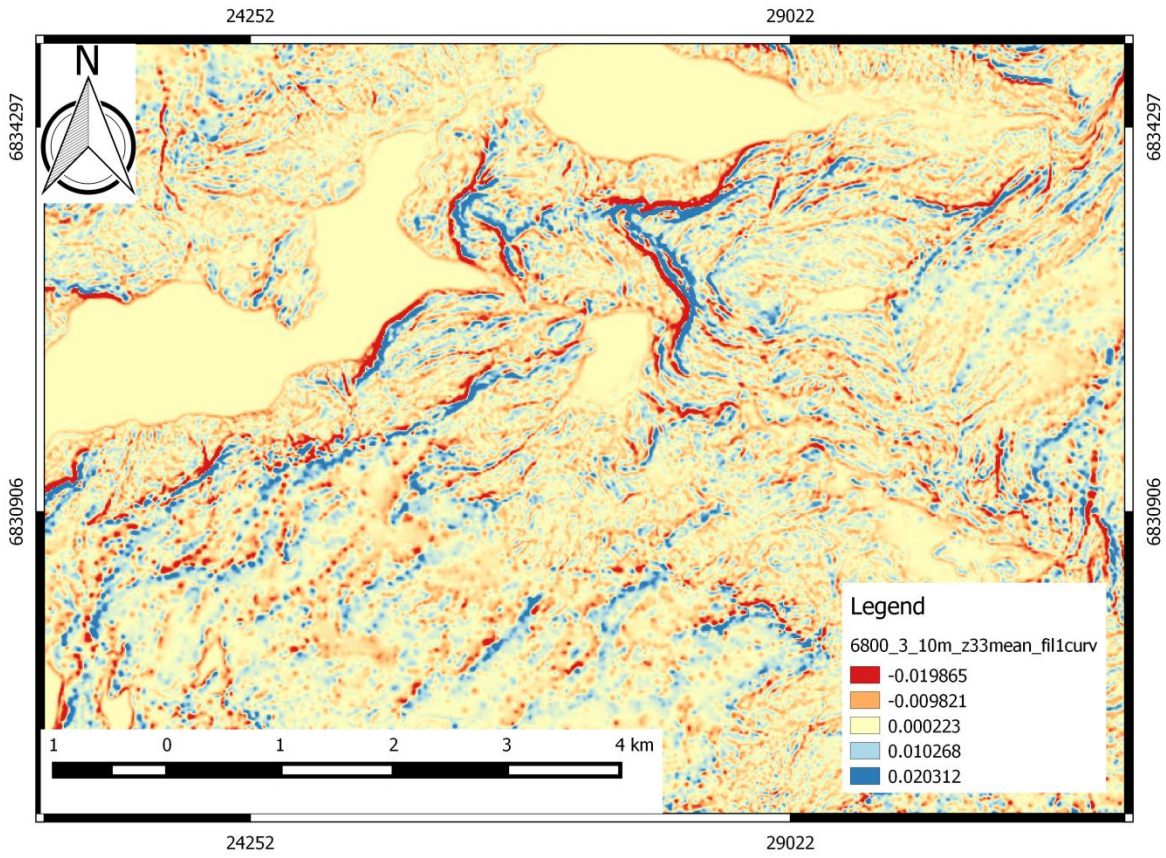


Figure 2.3 Mean curvature map. The map displays both negative and positive values, indicating both convex and concave areas.

2.3 The Gaussian image filter

Image filters are commonly used to remove noise by convolving the image with a mask (kernel, sliding window), in order to get a smoothed version of the image. For example a Gaussian mask is made out of elements which are part of a Gaussian function. The process of convolution brings the value of the pixels in closer harmony with the value of its neighbors (Lopes, 2012). Gaussian filters are a class of low-pass(McAndrew) filters and can be created using the following formula (two dimensional function):

$$f(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (15)$$

The key element is by far the standard deviation σ^2, which will dictate the values in the mask. The value from the center gets a bigger weight, and the weights decrease as the values are

located further from the center . By down weighting the distant pixels, one can enforce that, pixels which are closer to the center are more alike than the pixels which are further away(Forsyth, 2003).The Gaussian filter will have a blurring effect on the image, and it will yield similar results as the neighborhood averaging. If one wishes to spread out the blurring effect, a large standard deviation is needed(McAndrew), otherwise a small standard deviation will have little effect , because the center weights will be very small(Forsyth, 2003). The benefit of a Gaussian smoothing compared to other methods is that the resolution at which intensity changes are manifested can be chosen (Vernon, 1991). In other words, a mask can have different standard deviations, therefore different smoothing degrees.

2.4 Digital Morphology

The goal of computer vision is to segment images into meaningful objects. To achieve this, it is common to use neighborhood operations. These can be simply defined as adding or removing pixels from a binary image, according to a rule (Russ, 1998).Such a method is the mathematical morphology. This can handle both binary images and gray level images, using only a structuring element and a defined set of rules.

2.4.1 Structuring Element

A structuring element(Serra, 1986) is a small set which is used to probe the binary or gray level image. Structuring elements can have different shapes and sizes, but their dimensionality mostly depends on the dimensions of the image. Two dimensional images can only handle a two dimensional structuring element. Depending on the purpose, structuring elements can have the following shape: disk, hexagon, square, diamond, pairs of points (figure 2.4). In order to give a meaning, one must have information about the origin, and the size (Soille, 2013).

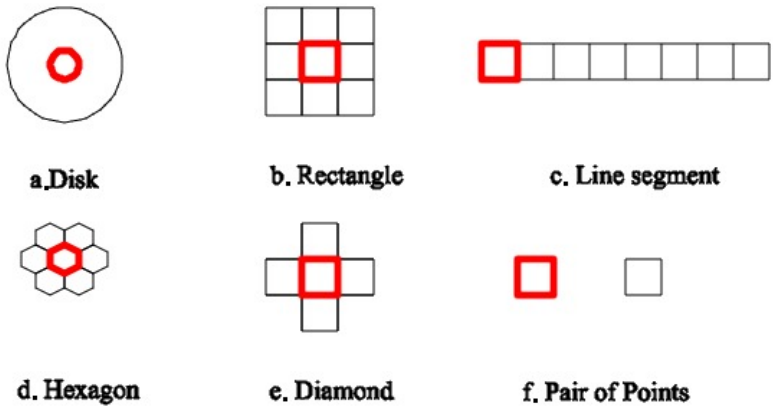


Figure 2.4 commonly used structuring elements in mathematical morphology. The red circle, rectangle, hexagon, represent the centers of the structuring elements.

In case of gray scale morphology, structuring elements can be flat (2D) or non-flat (n+1 dimensional)(Soille, 2013), and also some pixel positions may or may not be taken into account when the method is applied. The values within the structuring elements represent weights that will eventually be used in operations like erosion, dilation, opening and closing(Soille, 2013).

2.4.2 Gray scale morphology.

Erosion and dilation are the fundamental operations in morphological image processing, opening and closing are the result of combining erosion and dilation(Gonzalez and Woods, 2002).

The structuring element can be flat or non-flat and the image A can have any gray level value(Gonzalez and Woods, 2002).

The gray-scale dilation of image A by a structuring element $s(x, y)$, can be defined as

$$A \oplus s(x,y) = max\{A(x-x', y-y') + s(x',y') | (x',y') \in D_s\} \quad (16)$$

where D_s is the domain of s, and $A(x, y)$is assumed to equal $-\infty$ outside the domain of A (Gonzalez and Woods, 2002).

In other words this operation “thickens” or “grows” the objects of the image in the zones where the structuring element partially overlaps the image objects. The extent of the new objects

will be dictated by the shape of the structuring element(Gonzalez and Woods, 2002). Usually it has the effect of closing small holes within an image object.(Richards and SpringerLink, 2013)

On a digital elevation model, with a flat structuring element of size (70,70) this operation would have the effect of increasing the elevation values. Figure 2.5 has the hillshade of a section of an elevation model (top) and on the bottom the same section is subjected to morphological dilation. By increasing the gray values, the elevation model "thickens" and the valleys are filling up.

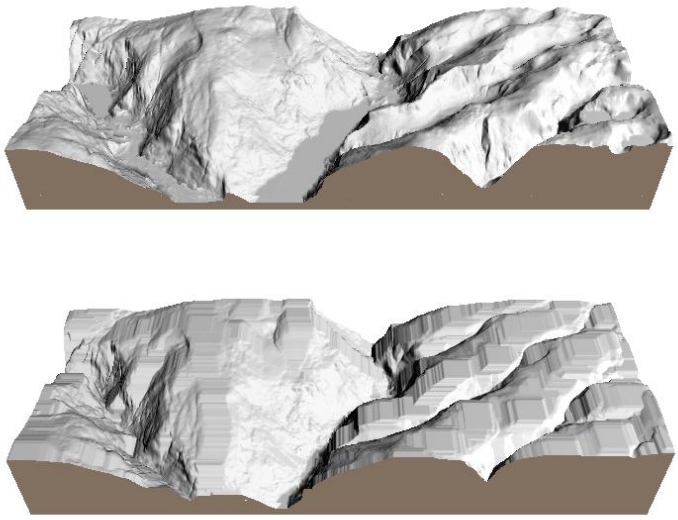


Figure 2.5 Morphological gray scale dilation. On the top a hillshade of a section of an elevation model is presented. On the bottom the morphological dilation of the section is shown. The overall values in the elevation model are increased, thus having a "thickening" effect on the elevation model.

Gray-scale erosion of image A by a structuring element $s(x, y)$ is defined as

$$A \ominus s(x, y) = min\{A(x - x', y - y') - s(x', y') | (x', y') \in D_s\} \quad (17)$$

where D_s is the domain of s, and $A(x, y)$is assumed to equal $-\infty$ outside the domain of A. Assuming a flat structuring element, the local minimum is taken over the pixel neighbors determined by the shape of Ds(Gonzalez and Woods, 2002)

As the name implies the operation has the effect of eroding thus “shrinking” the size of the image objects. In many cases it can be used to reduce ragged edges (Richards and SpringerLink, 2013) .

The effect of this operation on a digital elevation model would be that the elevation values would be reduced. Figure 2.6 illustrates how an elevation model changes when subjected to such an operation. Firstly, the valley bottoms are getting wider, due to the propagation of the local minimum, and secondly, the elevation of the mountaintops gets reduced.

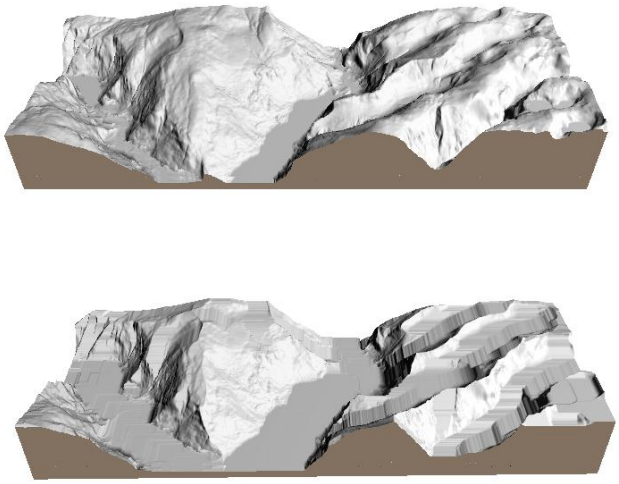


Figure 2.6 Morphological gray-scale erosion. Top of the image represents the unaltered hillshade of the elevation model. Bottom represents the elevation model subjected to gray-scale erosion. The overall effect of the operation is the reduction of gray level values, where valley bottoms become wider, and mountain tops get “thinned”.

Gray-scale erosion removes all structures that are not contained in the structuring element and shrink the other structures(Soille, 2013). Once image A is eroded by s, it is not possible to recover the initial image by erosion(Serra, 1986) . Image data is filtered out selectively, according to the size of the structuring element.

Matheron (1967)defined image opening as erosion followed by dilation, and closing as dilation followed by erosion , The formula for opening image A by a structuring element $s(x, y)$ is :

$$A \circ s = (A \ominus s) \oplus s \quad (18)$$

When subjecting an elevation model to gray-scale opening (figure 2.7) the mountain peaks will have a higher elevation, whereas valley bottoms will demonstrate little to no change. This in essence means that the elevations are exaggerated only for the mountain tops.

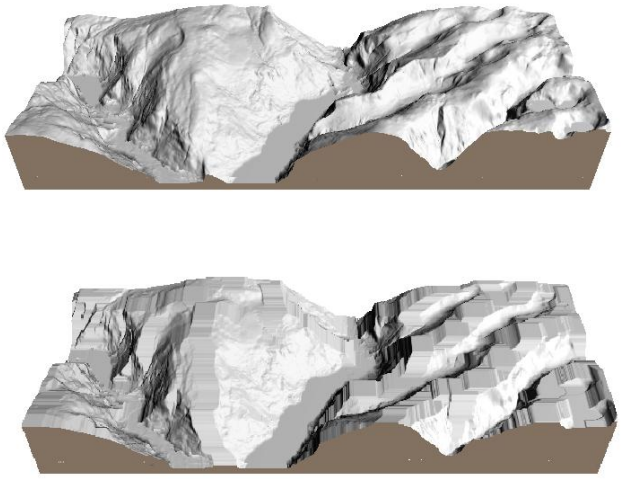


Figure 2.7 Gray-scale opening. Top: hillshade of the raw elevation model, hillshade of the gray-scale opened elevation model. The operation only affects the mountaintops while valley bottoms experience little to no change.

Similarly gray scale closing of image A by a structuring element $s(x, y)$ is, dilation followed by erosion:

$$A \cdot s = (A \oplus s) \ominus s \quad (19)$$

The operation represents the dilation of image A, followed by erosion. Suppose that image A is a digital elevation model. By applying a gray scale opening to the image, the structuring element would first find local minimums, and afterwards the local maximums, therefore the peaks in the digital elevation models are reduced.

Gray-scale closing applied on a digital elevation model will have the following effect (figure 2.8): areas of high elevation, such as mountain peaks will not be significantly changed, opposed to this, valley bottoms will be "widened".

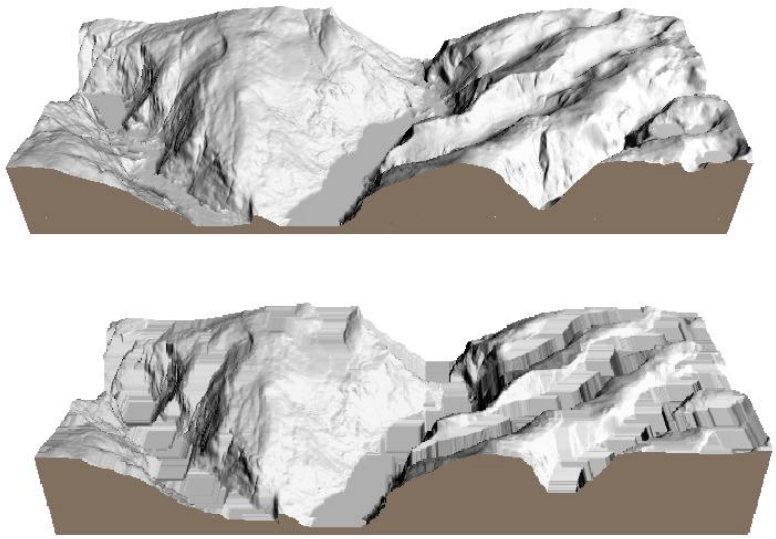


Figure 2.8 Gray-scale closing. Top: the raw digital elevation model. Bottom: the elevation model subjected to gray-scale closing. The mountain tops are left unchanged while the valley bottoms are widened.

2.4.3 (White) Top-hat / (Black) Top – hat transform

Top – hat transforms were introduced by Meyer (1979), for automatic screening of cervical smears. The white top-hat is the difference between the original image A and its opening, further noted as γ. Since the γ first takes a local minimum, and then a local maximum, the White – Top - Hat (WTH) image will always have greater or equal values than zero (Soille, 2000).

$$WTH(A) = A - \gamma(A) \quad (20)$$

The gray-scale opened image "thickens" the mountaintops. By differentiating this image from the original elevation model, greater values will appear on the ridgeline, therefore enhancing them. Figure 2.9 depicts the WTH transformed image (bottom), compared to the original elevation model (top)

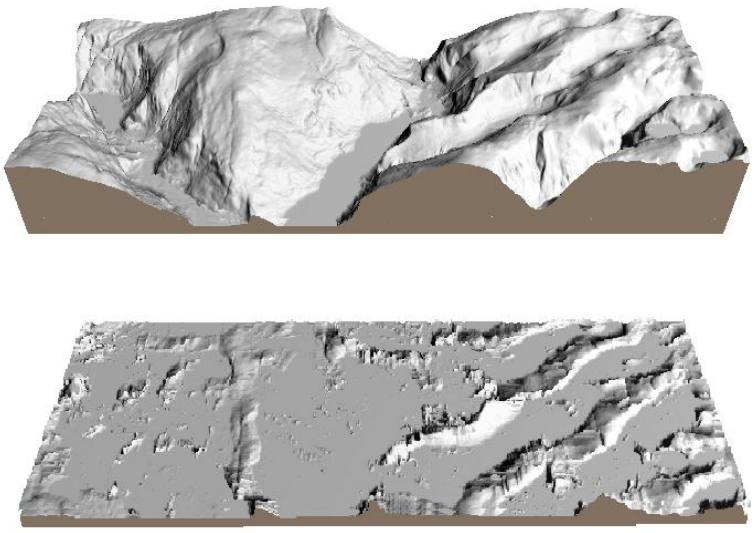


Figure 2.9 White – Top – Hat transformed image. This operation enhances ridgelines, by differentiating the original image from the gray – scale opened image.

Similarly the Black – Top – Hat (BTH) is calculated by differencing the original image A from the closed image, further noted as ϕ. Because the image is first dilated and then eroded, the values in the BTH image will always be smaller or equal to zero. These two operations are complementary operations (Soille, 2000).

$$BTH(A) = A - \phi(A) \quad (21)$$

When a digital elevation model is differentiated from the gray-scale closed version, the valley bottoms are enhanced. Because gray-scale closing does not have a strong effect on the mountaintops and ridgelines, these get eliminated from the image when the differencing is executed (figure 2.10).

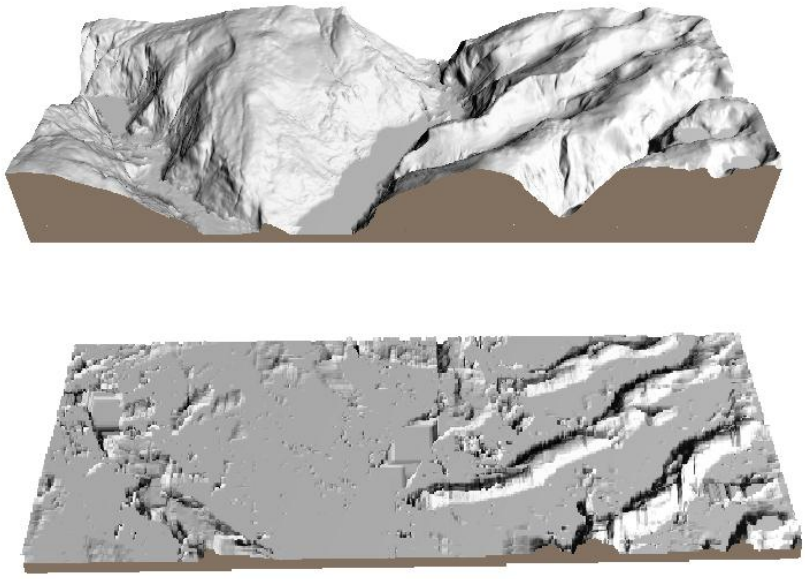


Figure 2.10 Black – Top – Hat transformed image. The original elevation model is differentiated from the gray – scale closed version, thus enhancing the valley bottoms.

2.5 Connected components labeling

Connected components labeling (Rosenfeld and Pfaltz, 1966) for a binary image is the procedure under which the unit in an image changes from a pixel level to a region level (Haralock and Shapiro, 1991). A subset of an image is considered to be connected if between any two points P and Q there exists a sequence of points $P = P_0, P_1, P_2 \ldots\ldots, P_{n-1}, P_n = Q$ of the subset in such a way that P_i is a neighbor of $P_{i-1}, 1 \leq 1 \leq n$ (Rosenfeld and Pfaltz, 1966).

Assuming a 3 x 3 window is used, one can calculate a 4 – connected region, in which case only the north, south, east and west pixels are taken into account, or an 8 – connected region, where besides the north, south, east and west, the northwest, southeast, southwest and southeast pixels are also taken into account (Haralock and Shapiro, 1991).

Figure 2.11 (left) depicts a binary image with circles in the foreground. By running a connected component algorithm(right), each individual circle is segmented into a region, thus being able to compare, analyze and use each circle individually.

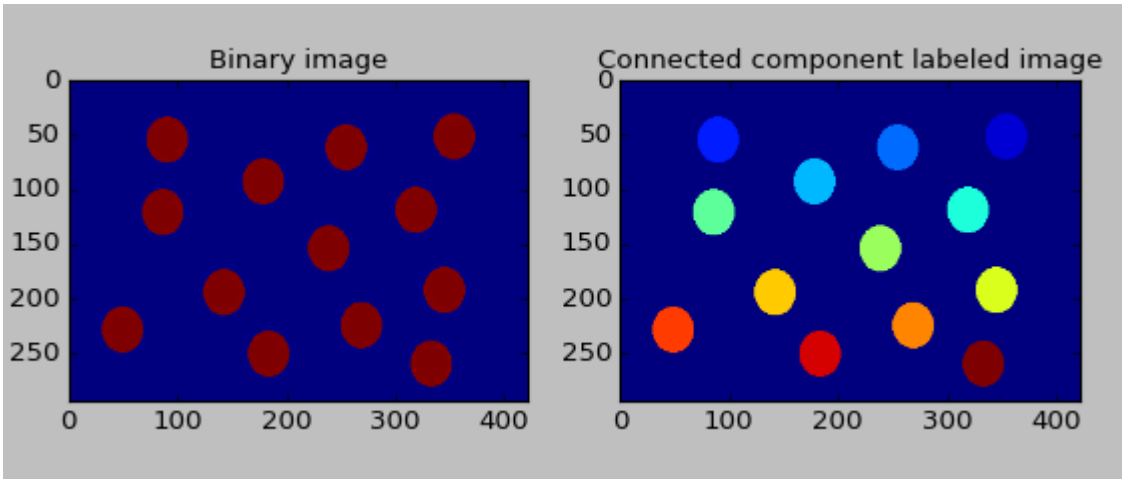


Figure 2.11Binary image with circles in the foreground (left), and connected component labeled image (right). Every circle becomes a separate entity, being able to use and analyze each circle individually.

2.6 Skeletons (Medial Axis Transform)

Introduced by Blum 1967, image skeletons have proven themselves over time, by being able to extract new shape descriptions from image objects. Used in computer vision applications, image skeletons have aided in various application such as: Biometric Identification of Iris(De Mira and Mayer, 2003), Ridge-based Fingerprint Recognition(Xie et al., 2005), Automatic grain boundary detection and grain size analysis (Heilbronner, 2000).

The purpose of skeletons is to reduce thick objects into one pixel wide image objects (Soille, 2013). Zhang and Suen (1984) presented an iterative method for thinning image objects in order to extract the skeleton of an image object. In the first step contour points are defined that have value 1, and at least one of the 8 neighbors with value 0. This is preceded by applying successive passes of two steps to these contour points.

In order to obtain the skeletons the following conditions must be satisfied:

- $2 \leq N(P_1) \leq 6$
- $A(P_1) = 1$
- $P_2 * P_4 * P_6 = 0$
- $P_4 * P_6 * P_8 = 0$

P9	P2	P3
P8	P1	P4
P7	P6	P5

Where $A(P_1)$ is the number of 0-1 patterns in the ordered set $P_2, P_3, \dots P_9$, which are the neighbors of P_1.

$N(P_1)$ is the number of nonzero neighbors of P_1, that is the sum of the neighbors of P_1.If any of the conditions are not satisfied, then the value of $A(P_1) = 2$,therefore P_1will not be deleted from the image.

In the second iteration the third and fourth condition will be changed to:

- $P_2 * P_4 * P_8 = 0$
- $P_2 * P_6 * P_8 = 0$

In this way, the sub iterations remove only the south-east boundary points and the north-west corners which don't belong to the ideal skeleton.

2.7 kNN(k Nearest Neighbor) classification

Used with success in in various computer vision applications from delineation of forest/non forest land use classes(Haapanen et al., 2004), to text categorization(Guo et al., 2004), the k Nearest Neighbor classification is a simple, yet time consuming method which can be used to cluster data. The assumption is that the pixels which are close to each other in the spectral space have a high probability of belonging to the same class. This supervised classification method assumes that an unknown pixel will be labeled by choosing the class from the training sample which is the most representative. (Richards and SpringerLink, 2013)

For this the distance needs to be computed between the unknown pixels and the training pixels. Assuming that there are k_i neighbors with labeled as class ω_i, which are the k nearest neighbors of pixel vector x. In this notation $\sum_{i=1}^{M} k_i = k$ where M is the number of classes. The discriminant function can be defined as following

$$g_i(x) = k_i \quad (22)$$

And the decision rule as $x \in \omega_i \; if \; g_i(x) > g_j(x) \; for \; all \; j \neq i$

In some cases it is beneficial to take into account the distance itself, in order to attribute every class a weight, making it easier to discriminate between classes with equal number of votes.(Richards and SpringerLink, 2013)

2.8 Multivariate Gaussian classification.

Statistical classification in general, has two approaches: parametric or non-parametric. A multivariate Gaussian classifier is a parametric classification method. It requires a probability distribution in order to estimate the parameters mean and standard deviation, and also a function in order to separate the classes and to cluster them(Geoff et al., 2010). In order to understand the theory behind it one must first understand the Bayesian Decision Theory. Considering $\omega_1, \omega_2 \ldots . \omega_c$ as a finite number of categories, x as a feature vector, the posterior probability $P(\omega_i|x)$ is equal to the product of the likelihood $p(x|\omega_i)$ and the prior probability $P(\omega_i)$ divided by the normalizing factor $p(x)$. In the case of unknown prior probabilities it is preferable to assume that all classes have the same prior probabilities (Duda et al., 2001)

$$P(\omega_i|x) = \frac{p(x|\omega_i) * P(\omega_i)}{p(x)} \quad (23)$$

Where the denominator is

$$p(x) = \sum_{j=1}^{c} p(x|\omega_i) * P(\omega_i) \quad (24)$$

What essentially the formula expresses is that the pixel represented by a d dimensional vector x, is a part of class $\omega_i$if $P(\omega_i|x)$ is the highest probability of the set. In order to further compute the posterior probability, one needs to choose a probability model for the class conditional density function $p(x|\omega_i)$. The most commonly used one is the multivariate normal distribution also called the Gaussian distribution(Richards and SpringerLink, 2013). The function has the following form:

$$p(x|\omega_i) = \frac{1}{\left((2\pi)^{d/2} * |\Sigma|^{1/2}\right)} * \exp\left(-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)\right) \quad (25)$$

Where

μ- is a d dimensional vector with the means of the classes

Σ- is a d x d dimensional covariance array between the classes

The covariance measures how two variables vary with respect to each other. In this sense the covariance matrix is used to summarize the covariance between the predefined classes. The matrix by definition is square and symmetric. The array also describes the degrees of similarity between features in a given class.(Dougherty, 2012)

The exponent of the density function represents the squared Mahalanobis distance. It is appropriate to use this when the variable scale differ and are correlated, but still approximately Gaussian Distributed(Dougherty, 2012). What the Mahalanobis distance does essentially is scale the absolute distance by their corresponding standard deviation therefore resulting in a more "probabilistic" measure of distance(Solomon and Breckon, 2011).

Essentially during the computation, Gaussian distributions are "built", similar to the ones in figure 2.12, in order to discriminate between classes, and all of the image pixels are then compared to these distributions in order to determine the probability of being part of a class.

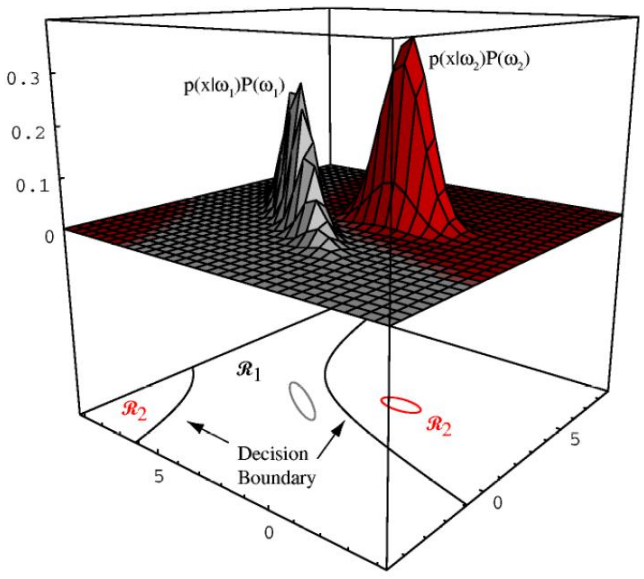


Fig 2.12 Two-dimensional two category classifier, the probability densities are Gaussian. The image also contains the decision boundaries, which represent the zones with equal probability of a pixel being classified in both class one and class two. Image credit Duda et al. (2001)

3. Methodology

The flowchart in figure 3.0 presents the methodology for classification of valley forms. For valley detection, as a first step, the elevation models are subjected to a primary Gaussian filtering. Afterwards morphological closing is applied to the filtered image. The closed image is subjected to a secondary Gaussian filtering prior to being subtracted from the original image. The resulting image is thresholded and using connected component labeling, small image details are removed. After this step the image skeletons are computed , splitted and filtered. The aforementioned image is used to define the classes for the kNN classification, where the thresholded image will be classified to create valley segments.
For the valley classification, the original elevation model is filtered with a small sigma filter, afterwards topographic parameters are calculated (slope and curvature). In order to train the classifier a training set is collected and the mean vectors and covariance matrices are calculated. As a final step, using the aforementioned parameters, the valley segments are classified

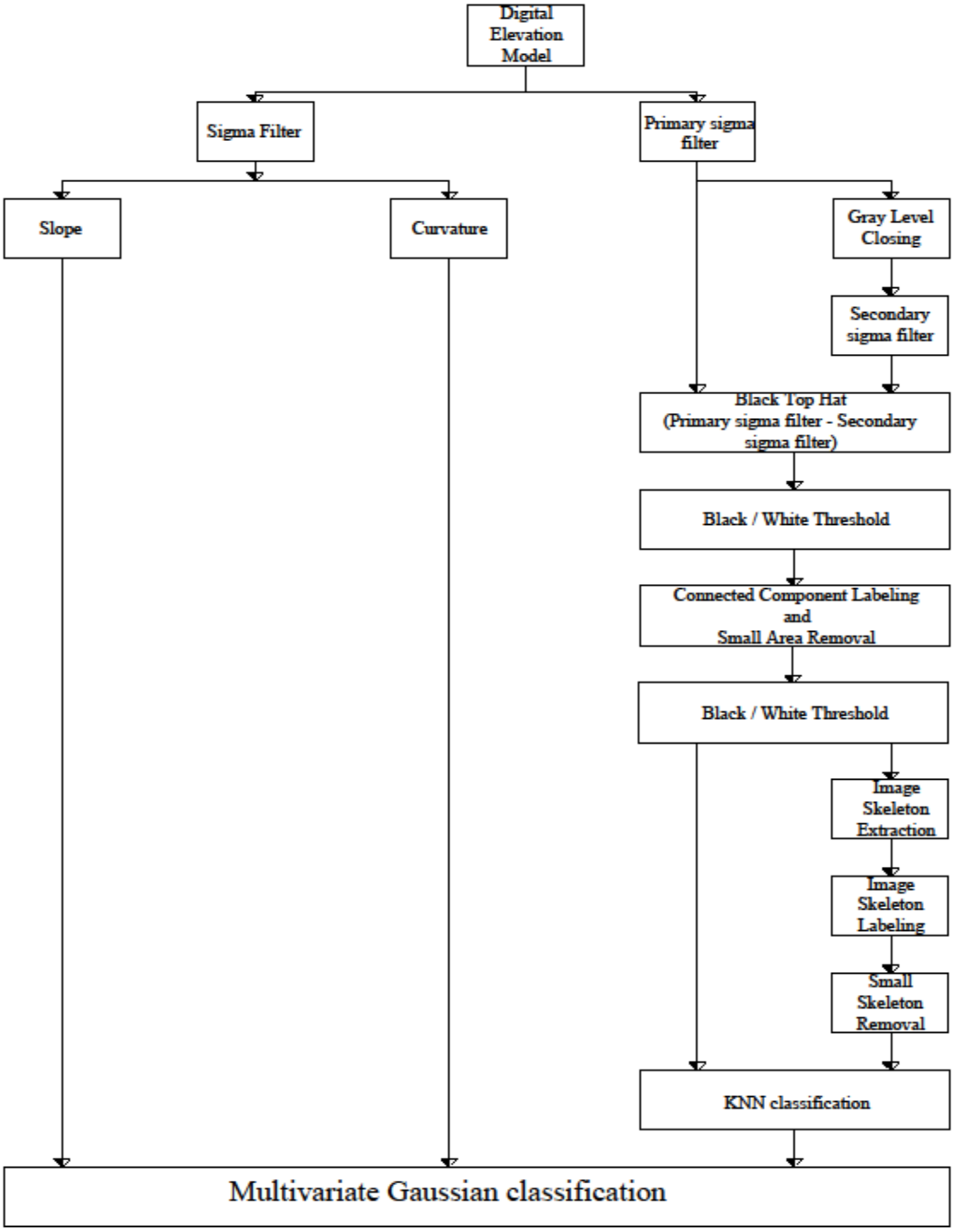


Figure 3.0 Methodology flow chart for automatic valley classification.

3.1 Processing platform

During the development of this method mainly two processing platforms were used: Python 2.79 and Quantum Gis 2.6 Brighton. Quantum Gis is an open source program licensed under the GNU General Public license. It is a powerful tool with core functions and plugins in order to visualize, manage, edit, analyze data and compose printable maps (QGIS_Development_Team, 2015).

Python is " an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together."(Python, 2015)

In order for python to work properly, the following libraries had to be installed: glob, PyQt4.QtCore, PyQt4.QtGui, qgis.core, qgis.utils, os, osgeo, numpy, osr, sys, math, PIL, math, matplotlib.pyplot, matplotlib.image, scipy, copy, skimage.morphology, numexpr, mpl_toolkits.mplot3d, scipy.ndimage. Additionally Qgis plugins are required such as Profile tool, and Qgistothreejs.

The processing and coding were executed on the open source platforms. The purpose of this was to make it open for anybody to use.

3.2 Data

3.2.1 Norwegian Mapping authority 10x10 m elevation model (Statkart 10)

The ten meter grid used in this thesis was created by the Norwegian Mapping Authority (Kartverket). Kartverket collates, systematize, manages and communicates public geographical information. In essence, it provides National Geodetic frame, positioning services, digital maps, printed maps, land registry, propriety information, place names, standards, primary ENC Service. (Kartverket, 2015c)

Norway can be found within 3 UTM zones, respectively 32, 33, 35. The Norwegian mapping Authority provides digital elevation models of the whole Norway in UTM zone 33. The DEMs give the heights above the sea level, in a grid with resolution of 10x10 meters. These models can be used for various applications from planning, to landscape analysis. The precision of the DEMs

varies from ±2 – 3 meters to ± 4 – 6 meters depending on the precision of the underlying data. Every Digital elevation model represents a 50x50 kilometer area. Al of the files have EUREF89 as a datum.(Kartverket, 2015a)

3.2.2 Norwegian Mapping authority 50x50 m elevation model (Statkart 50)

Similarly to the 10 m resolution, the 50 meter resolution grids were also created by the Norwegian Mapping Authority. The elevation models give the elevation above sea level with a 50x50 meter resolution, meaning that this elevation data contains 25 times less detail than the ten meter resolution grid. The underlying height information is the elevation data from N 50 Map Data and the road database. The precision of the data is ±4-6 meters depending on the detailing and precision of the underlying data. Every tile has the size of 100x100 kilometer. The data is in EUREF89 datum and the grid is produced in UTM zone 33.(Kartverket, 2015b)

3.2.3 Aster GDEM

Aster GDEM was jointly developed by the Ministry of Economy, Trade, and Industry from Japan and NASA. These digital elevation models can be downloaded free of charge from the Earth Remote Sensing Data Analysis Center of Japan and NASA's Land Processes Distributed Active Archive Center. The generation of these images includes processes as: extraction of corresponding points between images, pattern matching using normalized cross correlation, calculation of elevation using parallax, DEM stacking(Tachikawa et al., 2011).

The GDEM covers the Earth's surface between 83degrees North Latitude and 83 degrees South latitude. The vertical accuracy is estimated to be around 20 meters at the 95% confidence interval (Tachikawa et al., 2011).

3.2.4 SRTM elevation data

The Shuttle Radar Topography Mission was flown on the Space Shuttle Endeavour on flight STS-99 and was launched on 11 February 2000. It was cooperative mission between NASA and National Imagery and Mapping Agency (NIMA) and the Deutches Zentrum für Luft und Raumfahrt in Germany.

The elevation models were produced using Synthetic Aperture Radar (SAR) interferometry, on 80% of Earth's surface, 60 Degrees North latitude and 54 degrees South latitude (van Zyl, 2001). As for 2014 the SRTM V 3.0 one arcsec precision DEM is available for download. The precision of this grid is 30x30 meters, complementing the already available three arcsec precision DEM. The study will use the 3 arcsec precision elevation model with a grid size of approximately 90x90 meters(NASA, 2015).

3.3 DEM preprocessing and computation of slope aspect and curvature

Note: The steps in the methodology will be further exemplified on the Statkart 10 meter DEM. The zone is in south-western Norway near Bleia Storebotnen and Nærøyfjorden.

Because the digital elevation model has a high resolution, small local variations may arise when computing the slope and curvature, therefore a small sigma filter with $\sigma = 1$ will reduce some of the detail. This will attempt to bring the pixel values closer to their neighboring values (Lopes, 2012), and will result in a more homogeneous map. Figure 3.1 represents the digital elevation model that will be used to identify and classify the valleys. The elevation model originates from The Norwegian Mapping Authority and the grid size is 10x10 meters.

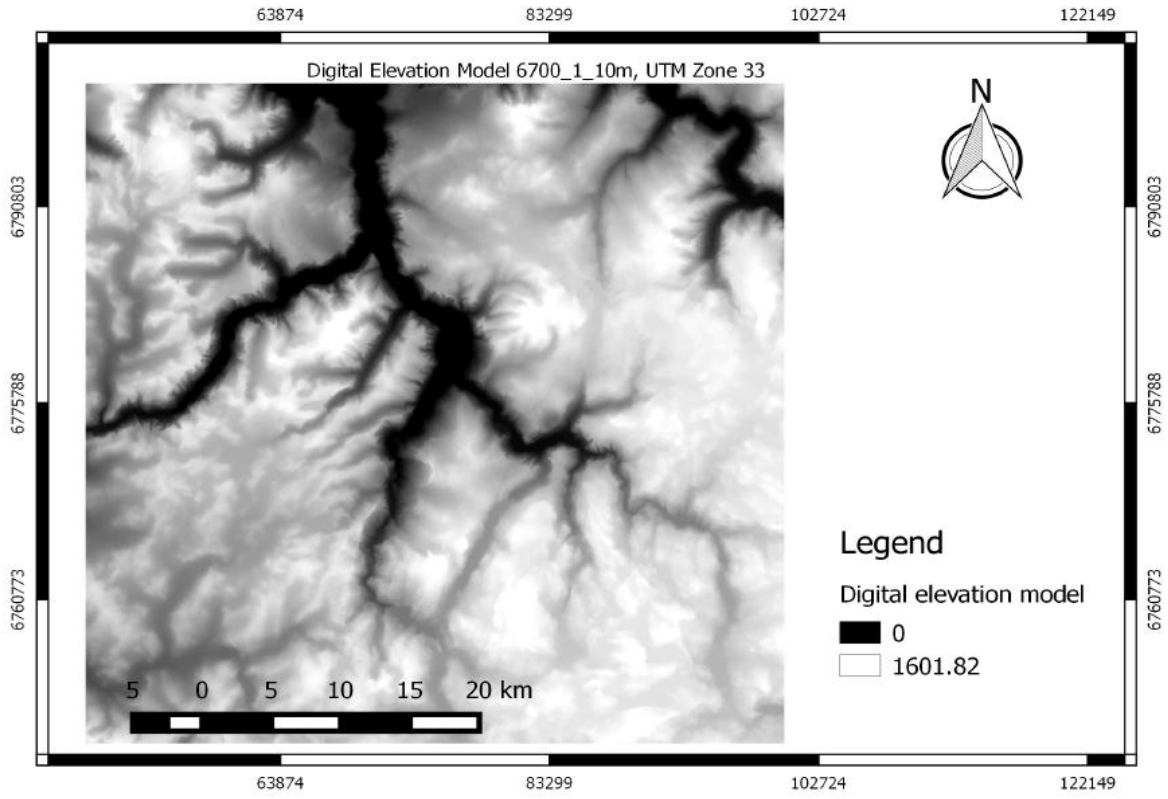


Figure 3.1 Digital elevation model further used to present the steps for valley classification. Approximate location is south-west of Norway near Bleia – Storebotnen and Nærøyfjorden .

3.4 Image filtering for the Black – Top – Hat transform

As a preprocessing step, the DEMs go under a larger sigma filter with the standard deviation of 15. Because the interest lies in the large landforms, the small and medium sized landforms are filtered out. Valleys are objects forged by nature, and may exhibit local variations like steep lines or local flats, which can have a negative impact on the results, therefore smoothing the topography eliminates some of these problematic areas.

Figure 3.2 displays the result of applying the Gaussian filter to the chosen digital elevation model. As expected the small details are blurred out from the image.

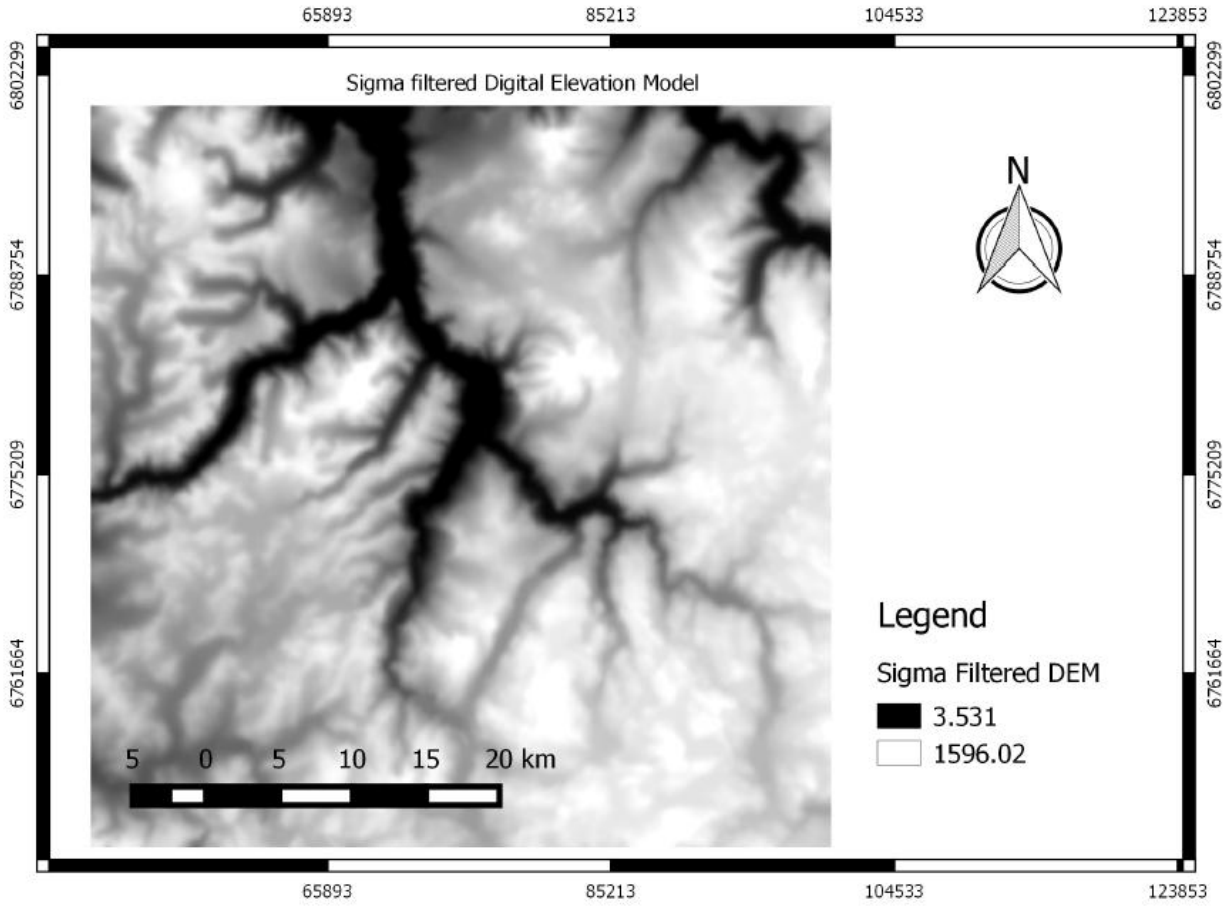


Figure 3.2 Digital elevation model filtered with a Gaussian σ=15 filter. As an effect the image appears blurred, and small details are eliminated.

3.5 Black –Top – Hat transform

When attempting to classify valleys, one must take into account the scale of the valleys that need to be identified. In the case of this master thesis, the objects of interest are large valleys; therefore the structuring element applied on the sigma filtered image needs to be large. The hypsography has a great influence also on the size of the structuring element. The optimal size of 201 x 201 pixels was chosen by trial and error, after analyzing more than 27 DEMs spread out throughout the whole Norway. Using the steps described in section 2.4.2 and 2.4.3, in the 201 x 201 window, the center cell is first substituted by the local maximum (gray level dilation). In the second step the center value is substituted with the minimum in the local window from the dilated image (gray level erosion), causing the valleys and cavities in the DEM to fill up. Figure 3.3 exhibits the result of the aforementioned steps. Furthermore, the resulting image

was also blurred by a Gaussian filter. The secondary sigma filter after the gray level closing is necessary to further smooth the edges. Its effect also enlarges the zones of interest, as in the zones where valleys are present get "expanded". By differentiating the original image by the closed DEM, large negative values appear where valleys are (Figure 3.4), thus obtaining the Black – Top – hat transformed image.

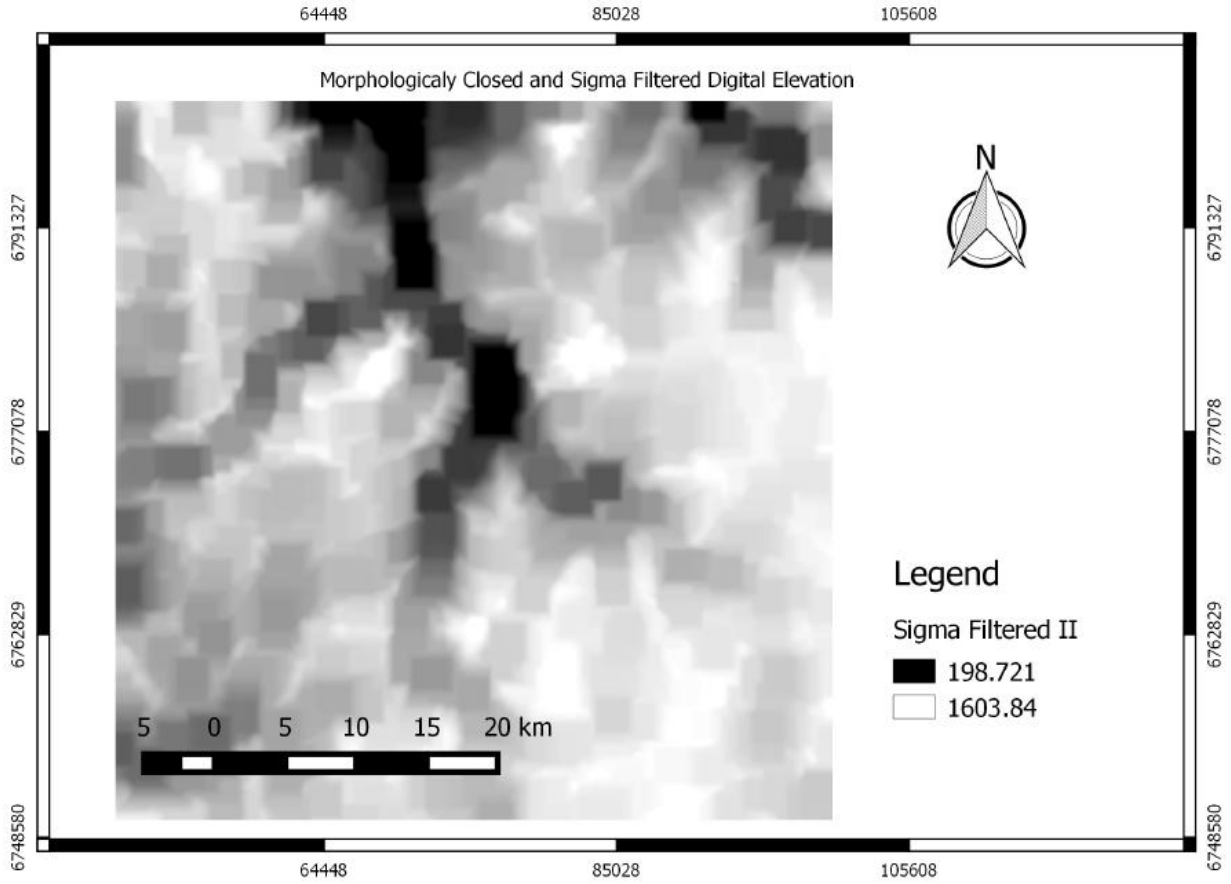


Figure 3.3 The morphological closing of the digital elevation model using a 201x201 structuring element. The image is also further smoothed by a Gaussian filter with σ=10. Blurring the image reduces the "edges" in the image, resulting in smoother shapes in the black – top – hat transform.

Figure 3.4 presents the valley networks in the digital elevation model, but also captures small cavities on the surface of earth. Deep valley bottoms will appear darker on the image, whereas points where variation is small will result in small negative values.

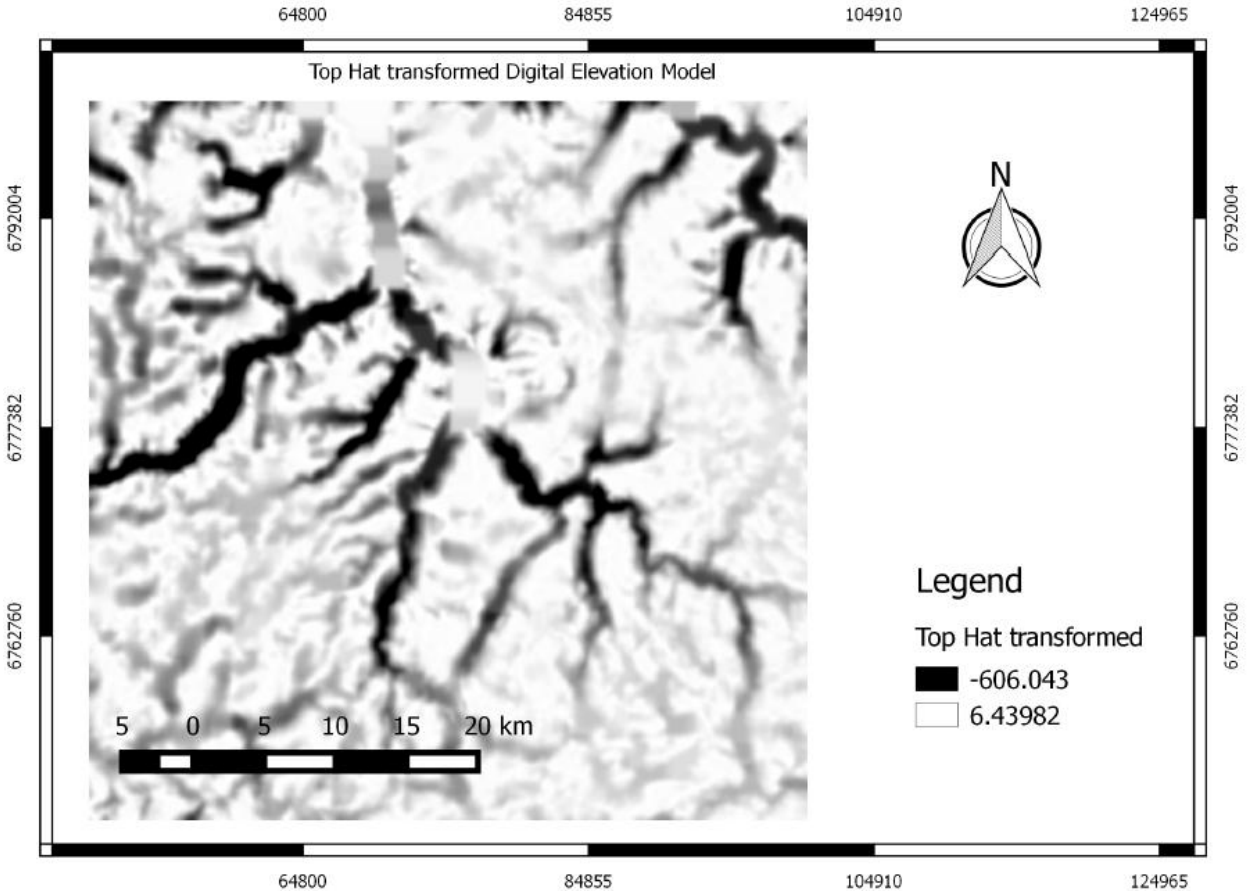


Figure 3.4 Top hat transformed image. The image results from the difference of the first filtered DEM from the closed image. The high negative values represent valleys. The darkest pixels represent the deepest points on the digital elevation model.

3.5 Thresholding

Because the Black – Top– Hat transform identifies valleys as well as cavities on the digital elevation models, one must find an optimal threshold value in order to eliminate areas with small elevation difference. As the Earth's surface can vary significantly, cavities with small elevation difference can appear in the Black – Top– Hat transformed image, therefore an optimal threshold of 100 meters was chosen. Any image pixel which has a lower value than -100 m will receive the value of 0, and the rest of the image will receive the value of 1. The result is a binary image with reduced detail.

Figure 3.5 represents the thresholded image, where small elements are present in the image which cannot be classified as valleys, because of their size. To have a more accurate classification, these small elements need to be filtered out from the image.

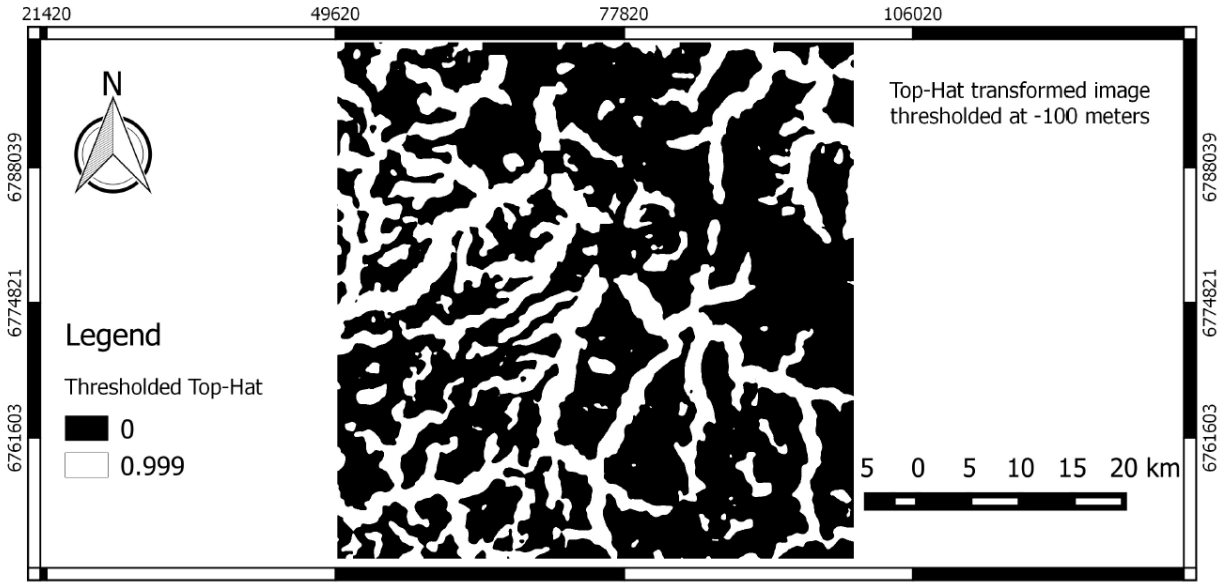


Figure 3.5 Black – Top – Hat transformed image, thresholded at -100 m. This step filters out cavities which are less than 100 meters deep, which results in a binary image.

3.6 Connected component labeling and small area removal

In order to get from a pixel level to an object level the binary image needs to be subjected to the connected component algorithm.

Figure 3.6 depicts the connected component image. Every group of foreground pixels acquires a unique value. Having attached labels to the binary data from the image, and looped through the objects, small entities which have fewer than 20000 pixels will be eliminated. The threshold value was chosen by trial and error, because the majority of valley networks have far more pixels than the threshold value. This reduces also the image detail, eliminating small isolated cavities from the DEM. Figure 3.7 shows the result of removing the small elements from the image. This image represents only the important valley segments,

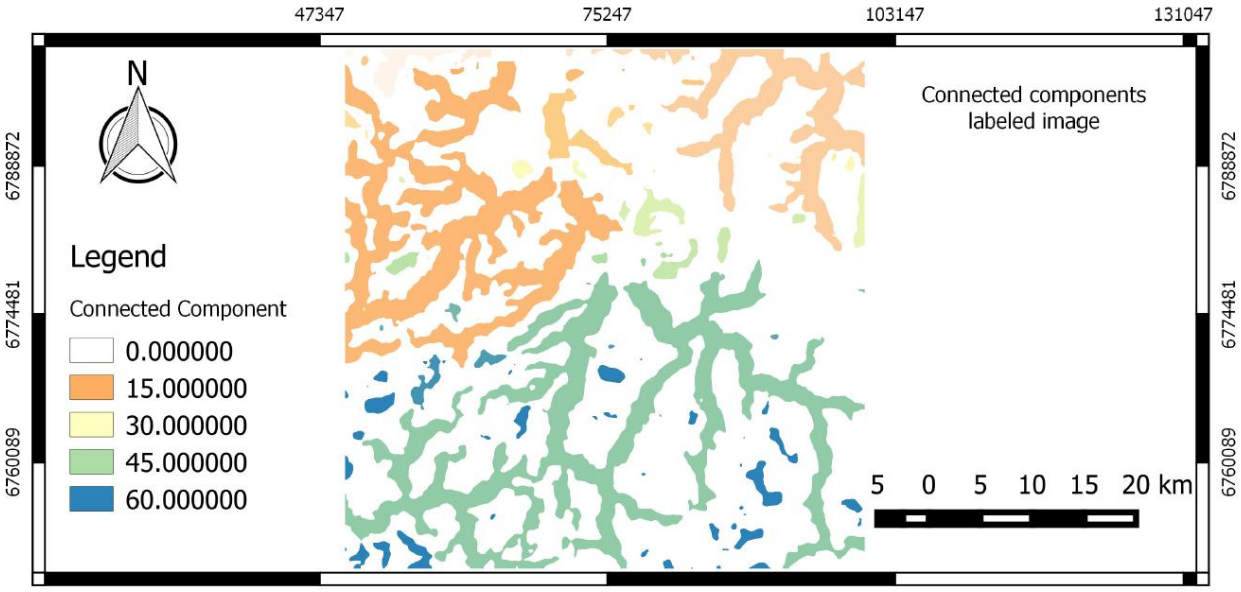


Figure 3.6 Connected Component Image. Image elements which are grouped and receive the same label creating image objects.

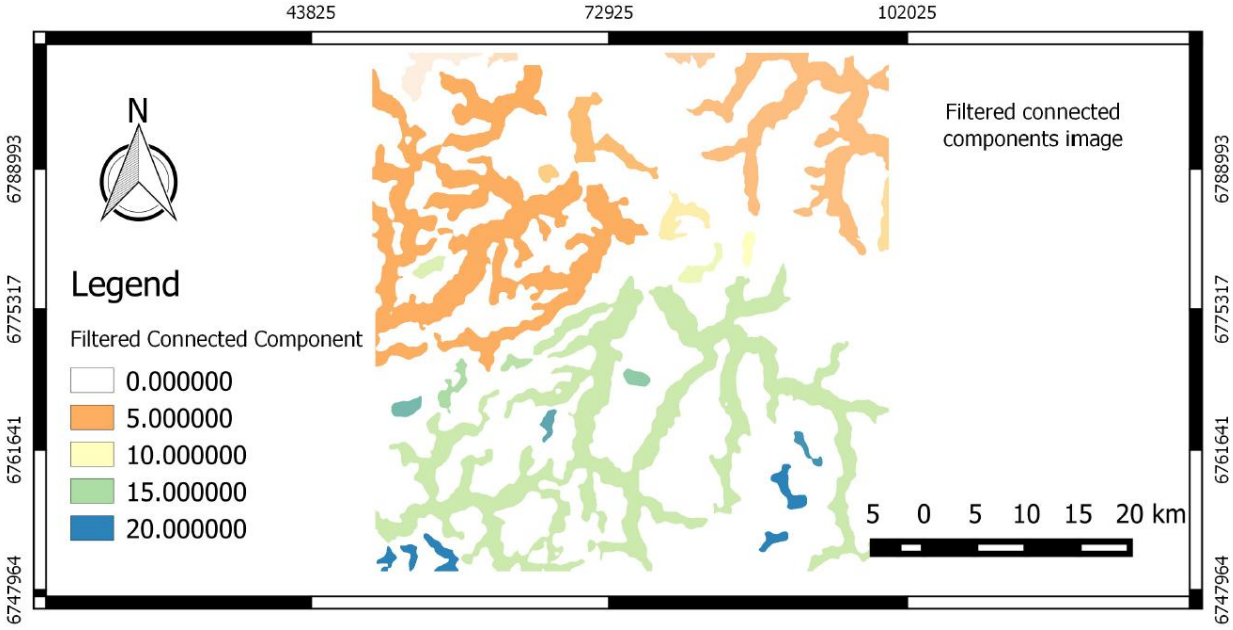


Figure 3.7 Connected components image filtered for small objects. Objects which have fewer than 20000 pixels are removed, resulting in an image with only the important valley segments.

3.7 Image skeletons

The medial axis transform of an object results in a center line of the object. In case of the thresholded images, the valley networks are reduced to lines in the image. The primary reason

why sigma filters are applied to the image is because image skeletons are highly sensitive to variations on the boundary(Geoff et al., 2010). Heavily smoothing the DEM helps create rounder objects. Having fewer corners ensures that every valley segment will have only one skeleton. Figure 3.8 depicts the image skeletons within the segmented image. The skeletons are present in the center of the objects. In order to get closer to the valley classification step, these skeletons need to be split into independent objects.

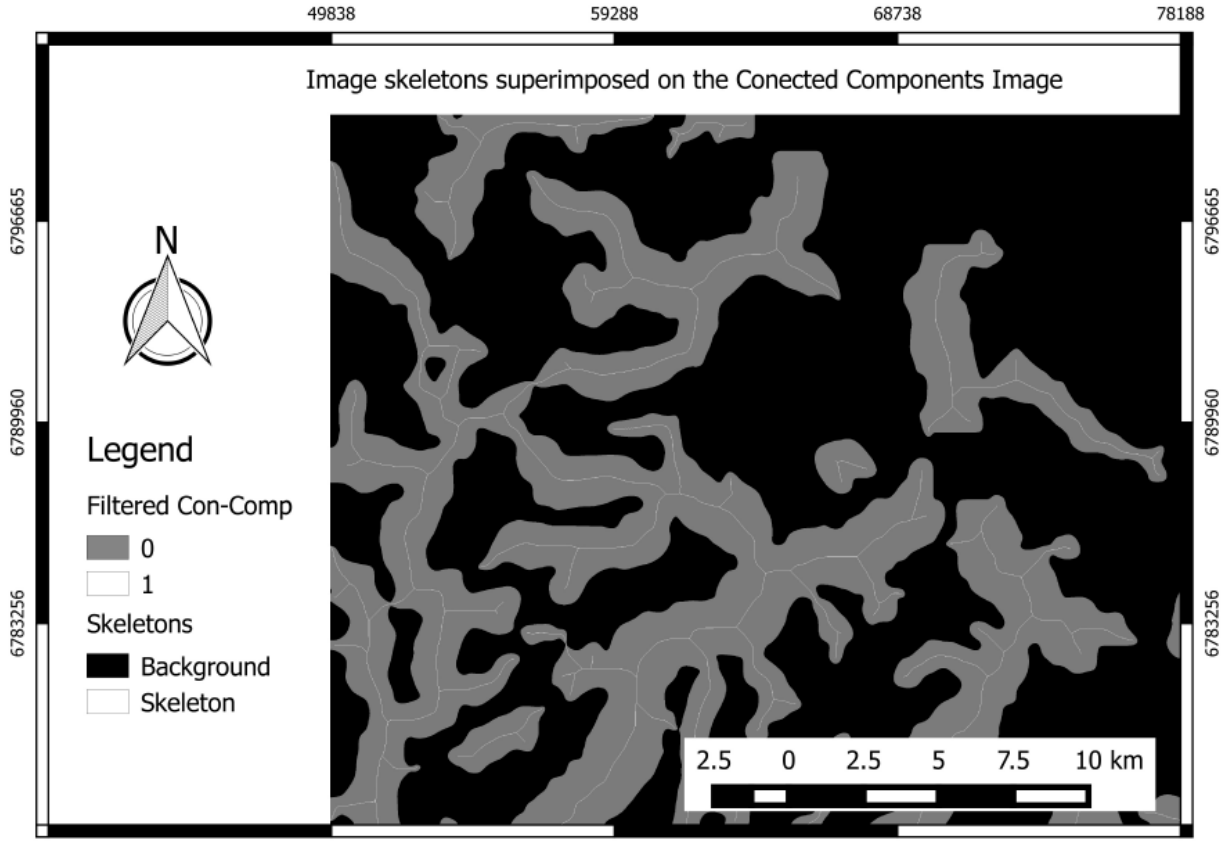


Figure 3.8 Image skeletons of the resulting filtered image. The skeletons are the lines which run in the center of the image objects.

3.8 Skeleton splitting and labeling

The skeleton splitting is a rudimentary algorithm, which splits the skeletons in the zone where multiple lines converge. Assuming a $5\ x\ 5$ window, and centering it in every image pixel which represents a skeleton, if the sum of the window is larger than 7, then the pixel is labeled as a point of convergence and deleted temporarily from the image. This will result in independent and unconnected lines. These are labeled with the connected component algorithm, and every

line which is smaller than 75 pixels is removed from the image. There is no interest in a skeleton with length less than this threshold, when using a structuring element of $201\ x\ 201$.

In the next step, the deleted pixels are added back to the image, and the process of splitting is executed again. Considering that some of the skeletons were deleted, there will not be as many convergence points as before, thus avoiding over segmentation. (consult appendix A for further details on the algorithm)

The last step represents the relabeling of the skeletons, resulting in image objects of length longer than 75.

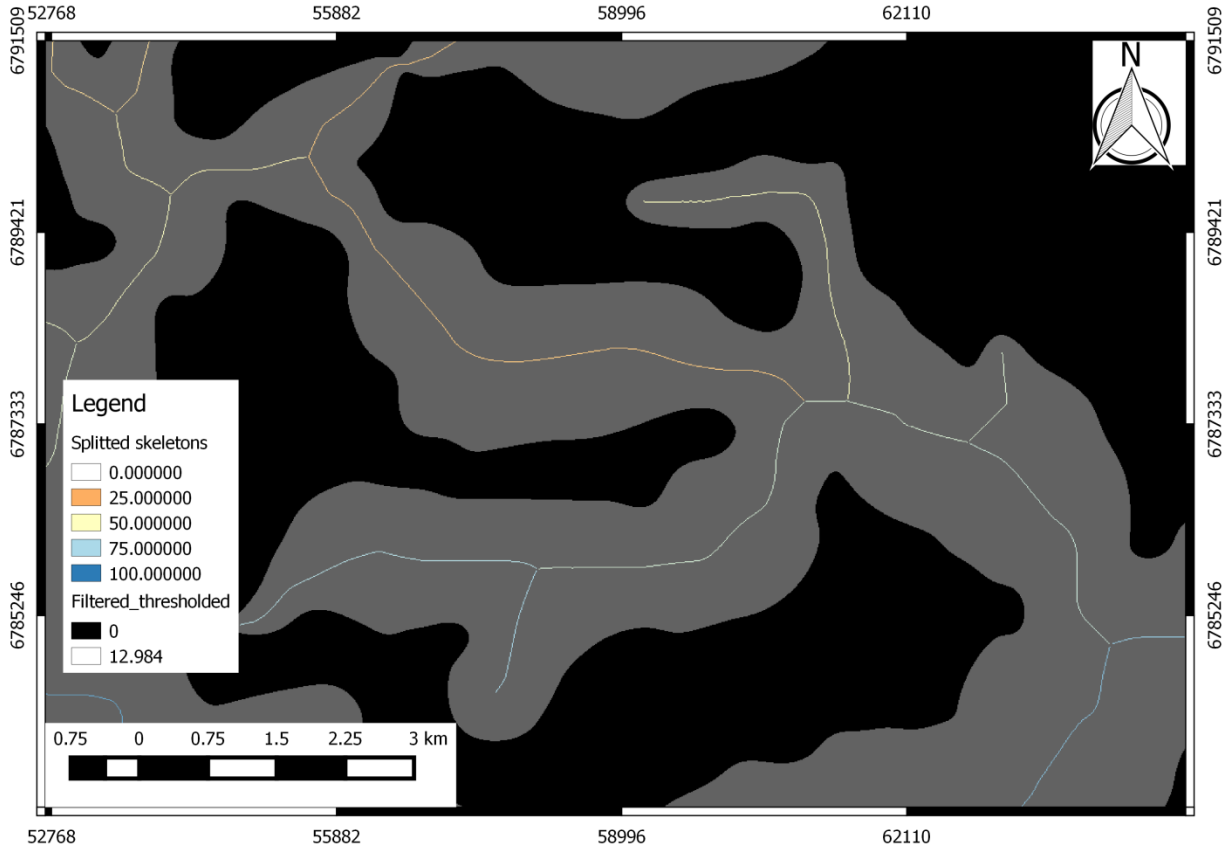


Figure 3.9 Split, labeled, and filtered image skeletons. Skeletons which are shorter than 75 pixels are automatically removed from the image

Figure 3.9 shows the skeletons which were split, labeled, and filtered by size. Essentially now every skeleton becomes one independent object.

3.9 The KNN classifier

There is one important assumption that needs to be made when applying this method: every skeleton is considered to be part of a valley, therefore every intersection of two valleys results in a new valley, which will be independent of the former two. Every small skeleton segment is an independent valley and therefore will become an independent object. Using this assumption, every pixel with value 1 from the image resulting in section 3.6, will be assigned to the closest image skeleton, segmenting the valley networks into independent valley segments. Figure 3.10 displays the result.

Because the labels are independent, if objects smaller than 20000 pixels occur, they can be optionally removed or reclassified to one of the larger classes.

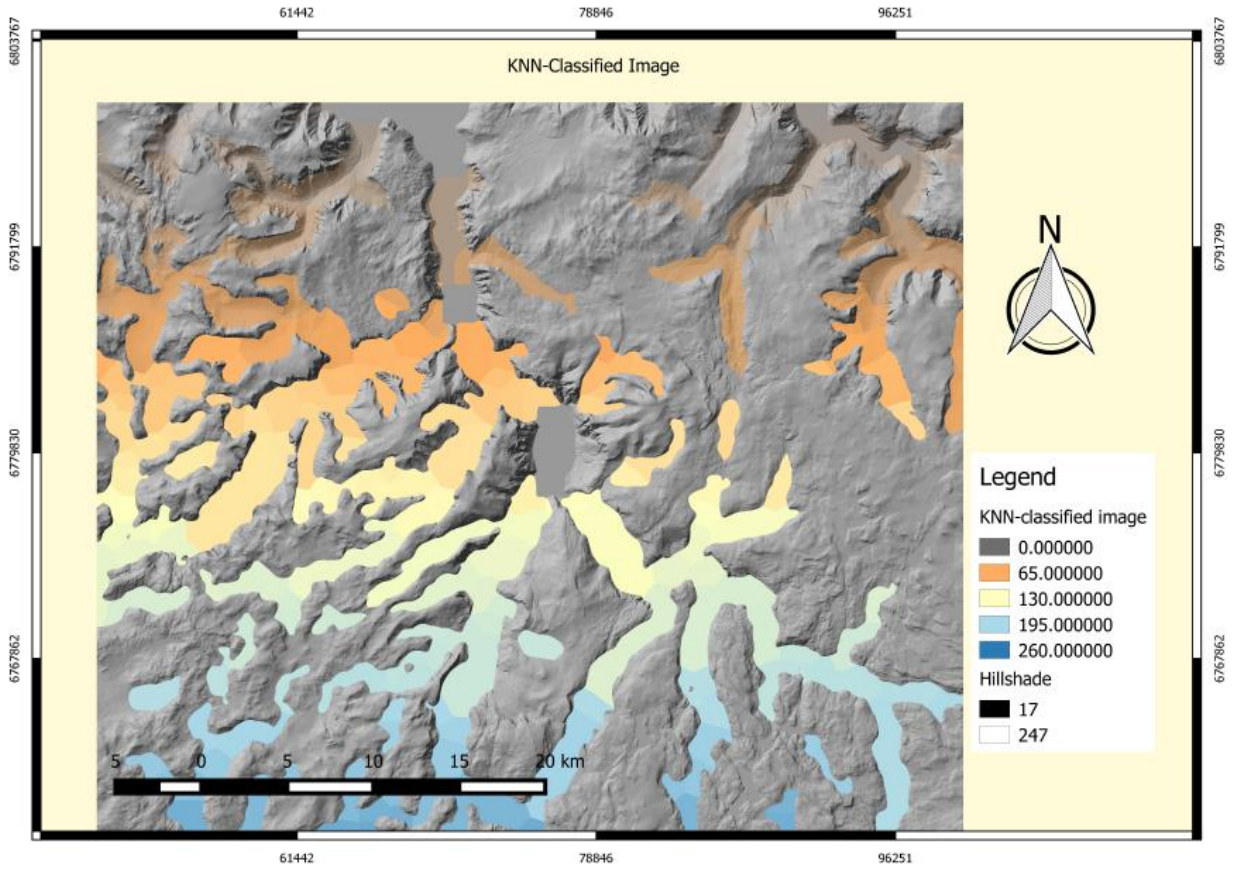


Figure 3.10 KNN classified image. Every pixel from the thresholded image is classified to the nearest skeleton, resulting in valley segments. The valley segments represent independent objects which will be further used in the multivariate Gaussian classification

3.10 Multivariate Gaussian Classification

In order to train the classifier, some samples were taken from the digital elevation models. These were selected from 17 DEMs spread across the whole Norway. From these 45 were U – Valleys, 35 were V – Valleys and 32 were Filled-Up-Valleys. For all of these the the number of pixels with slope under 5 degrees, between 5 and 30 degrees, the slope between 30 and 50, between 50 and 70 was calculated. Also the amount of pixels of with curvature under -0.0005, between - 0.0005 and 0.0005, and above 0.0005, was separated and scaled by their total sum. For further analysis referre to appendix I, II, III. The results from these samples allow to construct the covariance matrix and the mean vectors. Table 3.1 presents the means for every class and every feature that requires differentiating, and one can note that the means are slightly different from each other, therefore showing that the features used are different and aid the classification.

Seven features are used in the classification process and there are 3 class types, the covariance matrice will have a $7 \; x \; 7 \; x \; 3$ form. The covariance matrix for the valleys presented in table 3.2, table 3.3 and table 3.4.

Table 3.1 representing mean values of the samples taken. Table represents mean of slope under 5, interval (5,30), interval (30,50) and interval (50,70). It also contains curvature under -0.0005, interval (-0.0005, 0.0005) and above 0.0005.

Valley type	Mean values						
	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
U-Valleys	0.0280	0.7733	0.1754	0.0228	0.4959	0.0396	0.4645
V-Valleys	0.0445	0.7949	0.1420	0.0183	0.5002	0.0413	0.4585
Filled Up Valleys	0.1784	0.6636	0.1376	0.0201	0.4438	0.1551	0.4011

All covariance tables exhibit low values, because the extracted features are not overlapping resulting in little or no correlation between them. Low correlation also helps to compute the inverse for the matrix. If the values are correlated, the matrix tends to not be invertable.

Table 3.2. Covariance matrix for the seven features for V shaped valleys. The values are close to zero indicating that they are not correlated with each other

Covariance	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
S < 5	0.0002	0.0010	-0.0011	-0.0002	-0.0001	0.0001	0.0000
5 < S < 30	0.0010	0.0188	-0.0159	-0.0039	-0.0009	0.0010	-0.0002
30< S < 50	-0.0011	-0.0159	0.0138	0.0032	0.0008	-0.0009	0.0001
50< S < 70	-0.0002	-0.0039	0.0032	0.0009	0.0002	-0.0002	0.0000
Negative curvature	-0.0001	-0.0009	0.0008	0.0002	0.0001	0.0000	0.0000
No curvature	0.0001	0.0010	-0.0009	-0.0002	0.0000	0.0001	0.0000
Positive Curvature	0.0000	-0.0002	0.0001	0.0000	0.0000	0.0000	0.0001

Table 3.3 Covariance matrix for the seven features for U shaped valleys. The values are close to zero indicating that they are not correlated to each other

Covariance	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
S < 5	0.0005	0.0014	-0.0016	-0.0003	-0.0001	0.0001	-0.0001
5 < S < 30	0.0014	0.0088	-0.0084	-0.0017	-0.0003	0.0005	-0.0002
30< S < 50	-0.0016	-0.0084	0.0083	0.0017	0.0003	-0.0005	0.0002
50< S < 70	-0.0003	-0.0017	0.0017	0.0004	0.0001	-0.0001	0.0001
Negative curvature	-0.0001	-0.0003	0.0003	0.0001	0.0000	0.0000	0.0000
No curvature	0.0001	0.0005	-0.0005	-0.0001	0.0000	0.0001	0.0000
Positive Curvature	-0.0001	-0.0002	0.0002	0.0001	0.0000	0.0000	0.0001

Table 3.4 Covariance matrix for the seven features for Filled up valleys. The small values suggest that the features are not correlated between them

Covariance	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
S < 5	0.0393	-0.0281	-0.0097	-0.0016	-0.0183	0.0372	-0.0189
5 < S < 30	-0.0281	0.0359	-0.0064	-0.0013	0.0141	-0.0283	0.0141
30< S < 50	-0.0097	-0.0064	0.0137	0.0024	0.0036	-0.0078	0.0042
50< S < 70	-0.0016	-0.0013	0.0024	0.0004	0.0006	-0.0012	0.0007
Negative curvature	-0.0183	0.0141	0.0036	0.0006	0.0087	-0.0177	0.0089
No curvature	0.0372	-0.0283	-0.0078	-0.0012	-0.0177	0.0358	-0.0182
Positive Curvature	-0.0189	0.0141	0.0042	0.0007	0.0089	-0.0182	0.0092

After training the classifier, every valley section is used separately as a mask, and the slope and curvature is extracted from the maps. The values are divided into the previously mentioned groups and are scaled by the number of total pixels in the object. The process is repeated for every object in every DEM.

4. Results

4.1 Results Statkart 10 m grid

The most important feature in the identification step is the size of the structuring element. If the structuring element is too large, small valleys will be filtered out. If the structuring element is too small, the wide valleys will not be detected. Even with a structuring element which satisfies both conditions, information from the sides of the valleys will be lost, because it is seldom that the size of a valley matches exactly the size of the structuring element. The size can be adapted if the algorithm is run locally, but it is easy to select settings which fit one zone, but do not fit another. Testing the structuring element size on many elevation models ensures that overfitting does not occur.

Judging from the digital elevation models, u – shaped, and many filled up u – valleys tend to be wider than v – valleys, therefore details regarding curvature and slope are lost. This can have a negative impact in the final classification.

Figure 4.0 shows exactly how much detail is used from the filled up valley sides. A cross section is presented of filed up valley, and the black vertical lines exhibit what the algorithm identified . One third of the information is not taken into account. Even though there is a 30% information loss, the defining characteristic of these valleys isthe flat bottoms, which help to successfully differentiate from other types of valleys

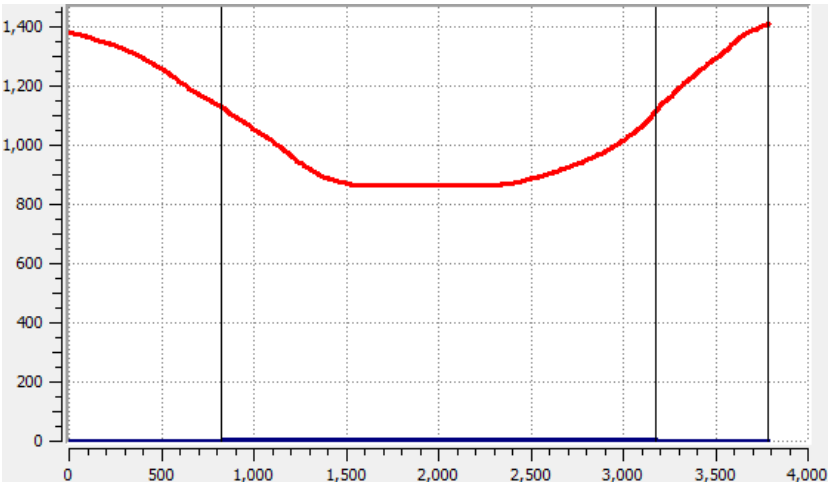


Figure 4.0 Actual valley size, versus the valleys identified by the Black – Top – Hat transform.The two vertical lines in the middle represent the width of the valley identified by the Black – Top – Hat transform. It can be seen that roughly 1/3rd of the data is not taken into account.

Figure 4.1 on the other hand shows a cross section of a u – valley. The valley sides tend to have a steeper slope then in case of the v– shaped valleys. Due to of the large sigma filter and the thresholding limit, usually the steepest parts from the valley sides are not identified, as figure 4.1 points out. In theory the average slope in a u – valley should be above 30 degrees, in reality because not all of the valley sides are captured, the average steepness is closer to the steepness of the v – valleys.

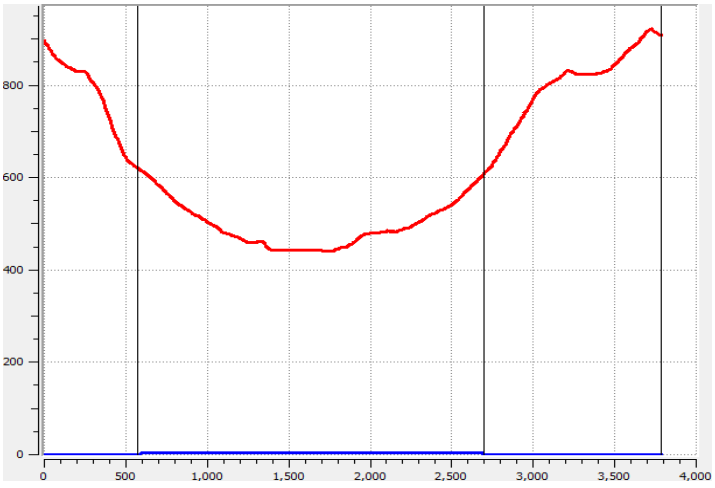


Figure 4.1 Cross section of u – valley, and cross section of valley identified by the Black – Top – Hat transform, bounded by the middle vertical lines. It can be seen that the identified segment is smaller, and the steep sides of the valleys are left out.

V shaped valleys however, are narrower, well identified. Even though a lot of detail is lost, as pointed out in figure 4.2 the steepest parts are captured by the algorithm.

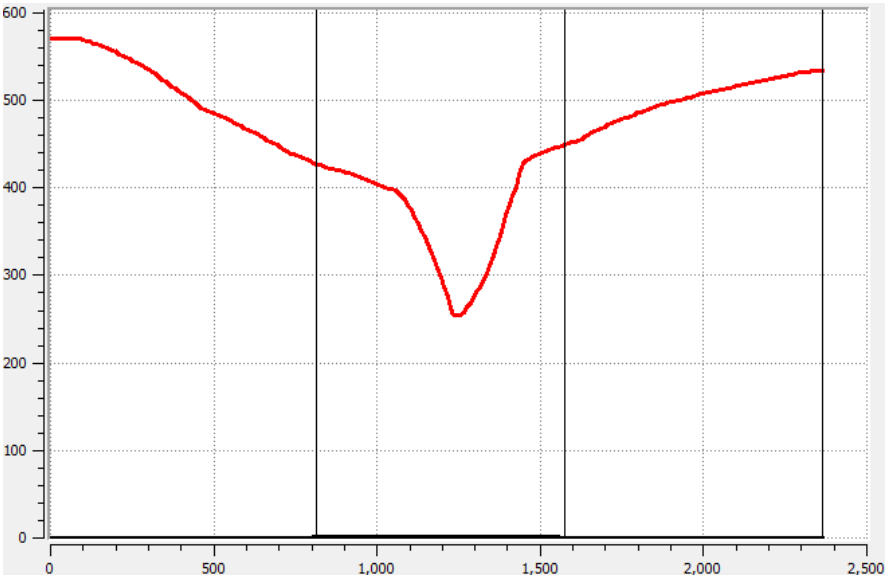


Figure 4.2 Cross section of a V valley. The middle vertical lines represent the section identified by the Black – Top – Hat transform. Even though details are lost during the identification process, the typical signature of this type of valley is not influenced to much.

An additional condition was added to the filled up valleys if it contains more than 5% pixels with elevation of 0, then it is automatically reclassified as a Fjord. The 5% threshold is selected because of the converging skeletons and the KNN classifier. Assuming a valley which converges with a fjord,some of the pixels classified to the valley will be part of the fjords. These misclassified pixels are near the convergence(figure 4.3).

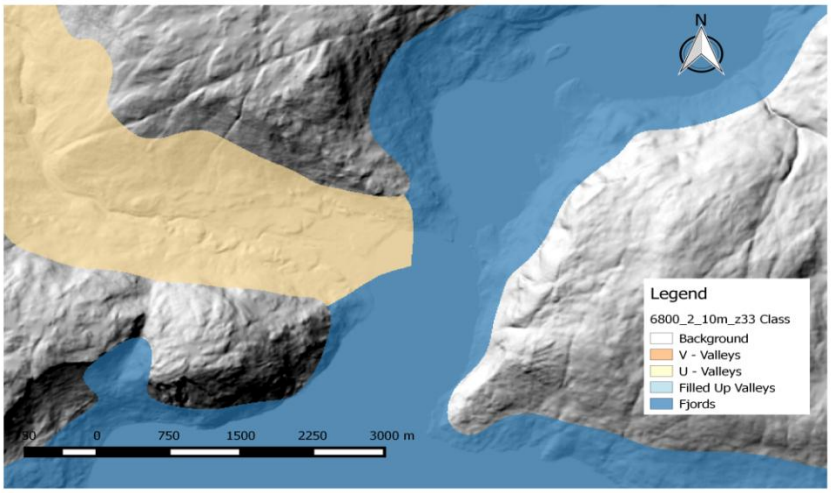


Figure 4.3 Converging U valley with fjord. Because of the kNN classification, pixels with 0 elevation get assigned to the U valley. If no tolerance is accepted, this valley would be classified as a fjord. Therefore a 5% tolerance in 0 elevation values is accepted.

The ease of differentiating between the classes arises from each class having an individual characteristic in feature space. Figure 4.4 displays the result of the method when applied to Statkart 10 meter resolution DEM. Furthermore, figure 4.5 depicts that the additional condition for the fjords, manages to separate the filled up valleys from the fjords. In both cases the valleys are identified and classified, with success. Lofoten area exhibits a different topography then other parths of Norway, with steep mountainous islands with sharp peaks, and carved cirques(Nordgulen et al., 2006) (figure 4.6), nevertheless the algorithm produces satisfactory results.

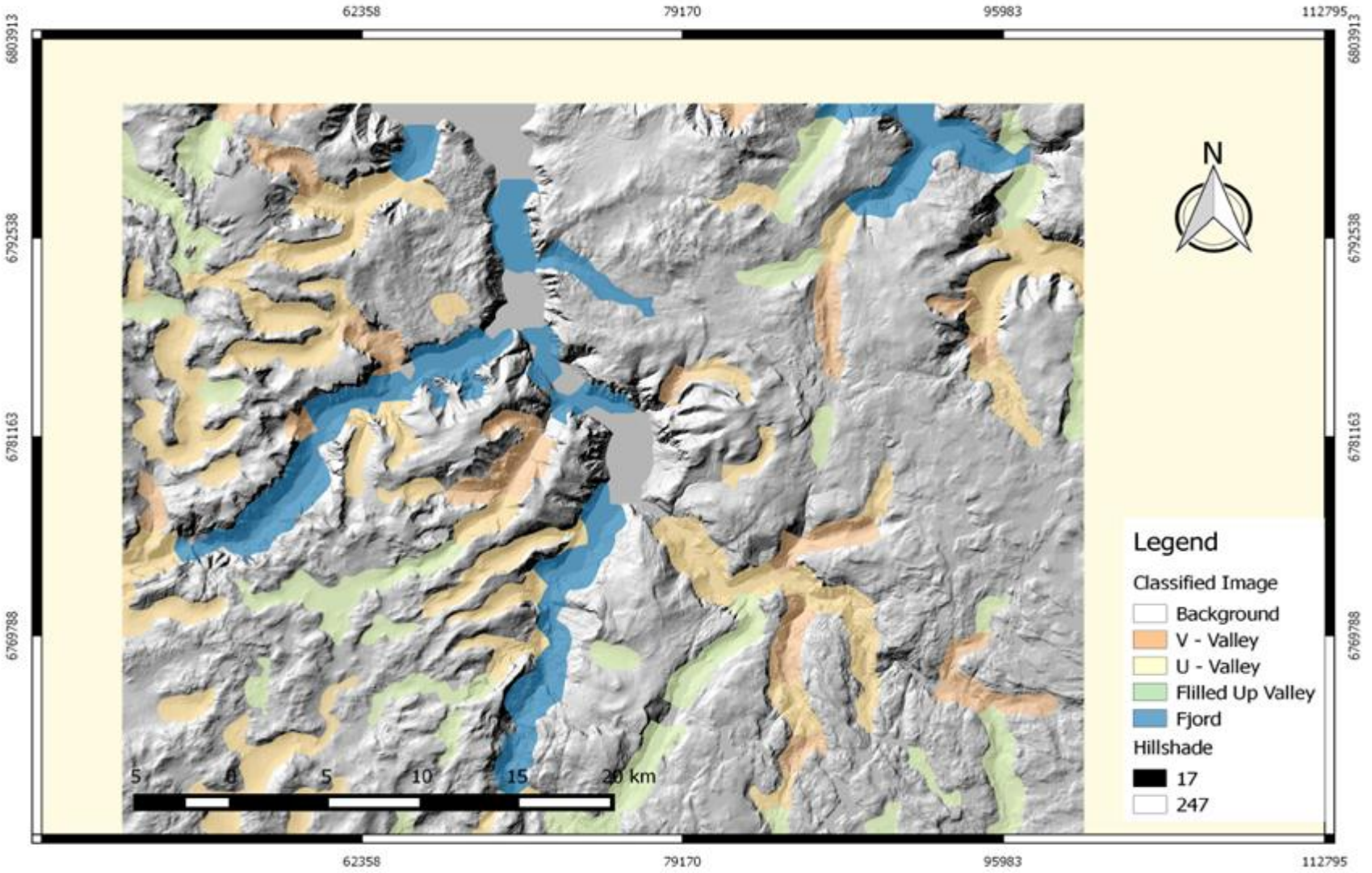


Figure 4.4 Multivariate Gaussian classification using the valley segments from section 3.7, and the geometric parameters of the terrain. Four main categories can be seen. u-valleys, v-valleys , Filled up Valleys and Fjords.

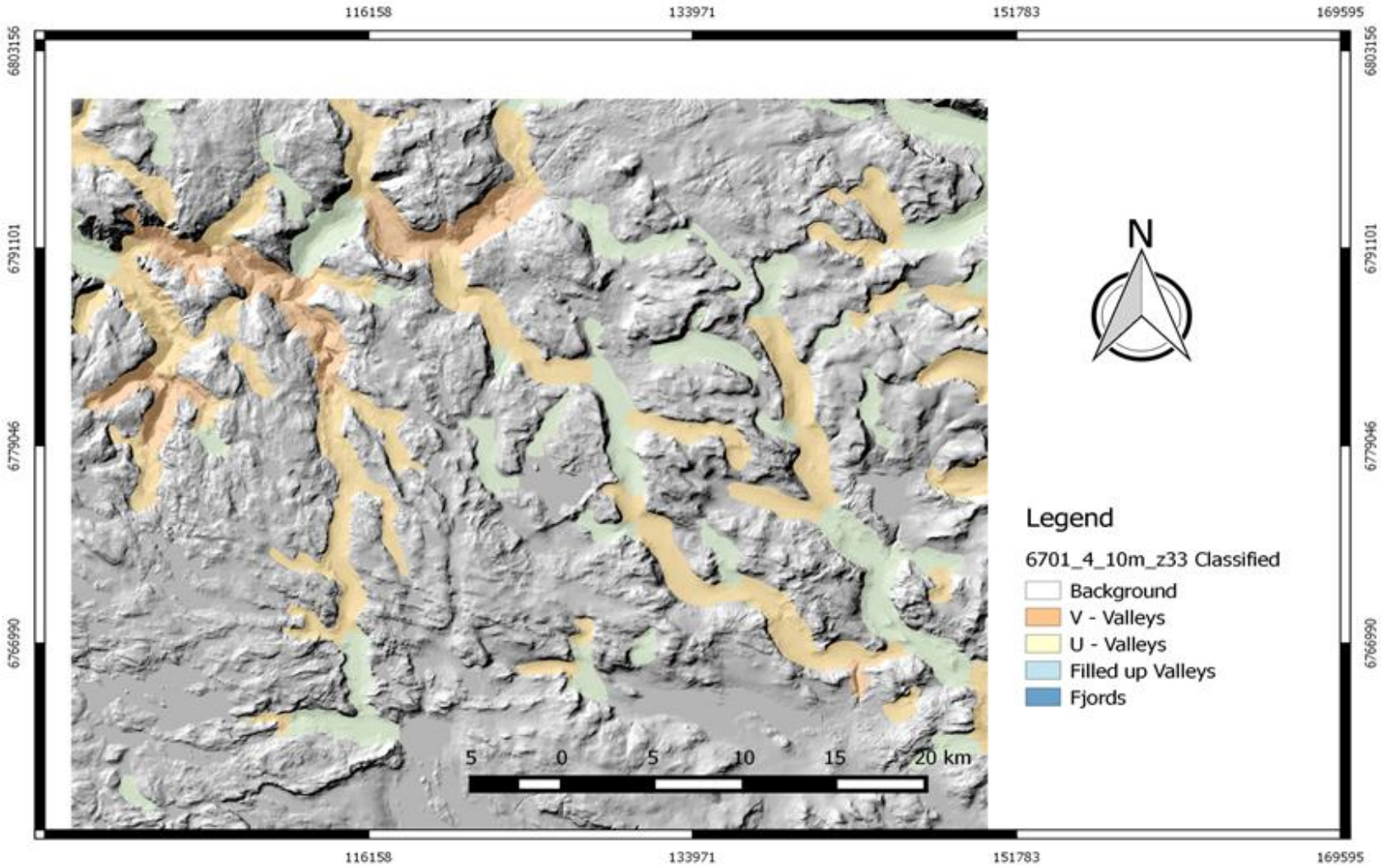


Figure 4.5 Result of the classification process. The algorithm manages to identify all three types of valleys v–valleys, u – Valleysand filled up valleys. Approximate location Lærdal, Norway

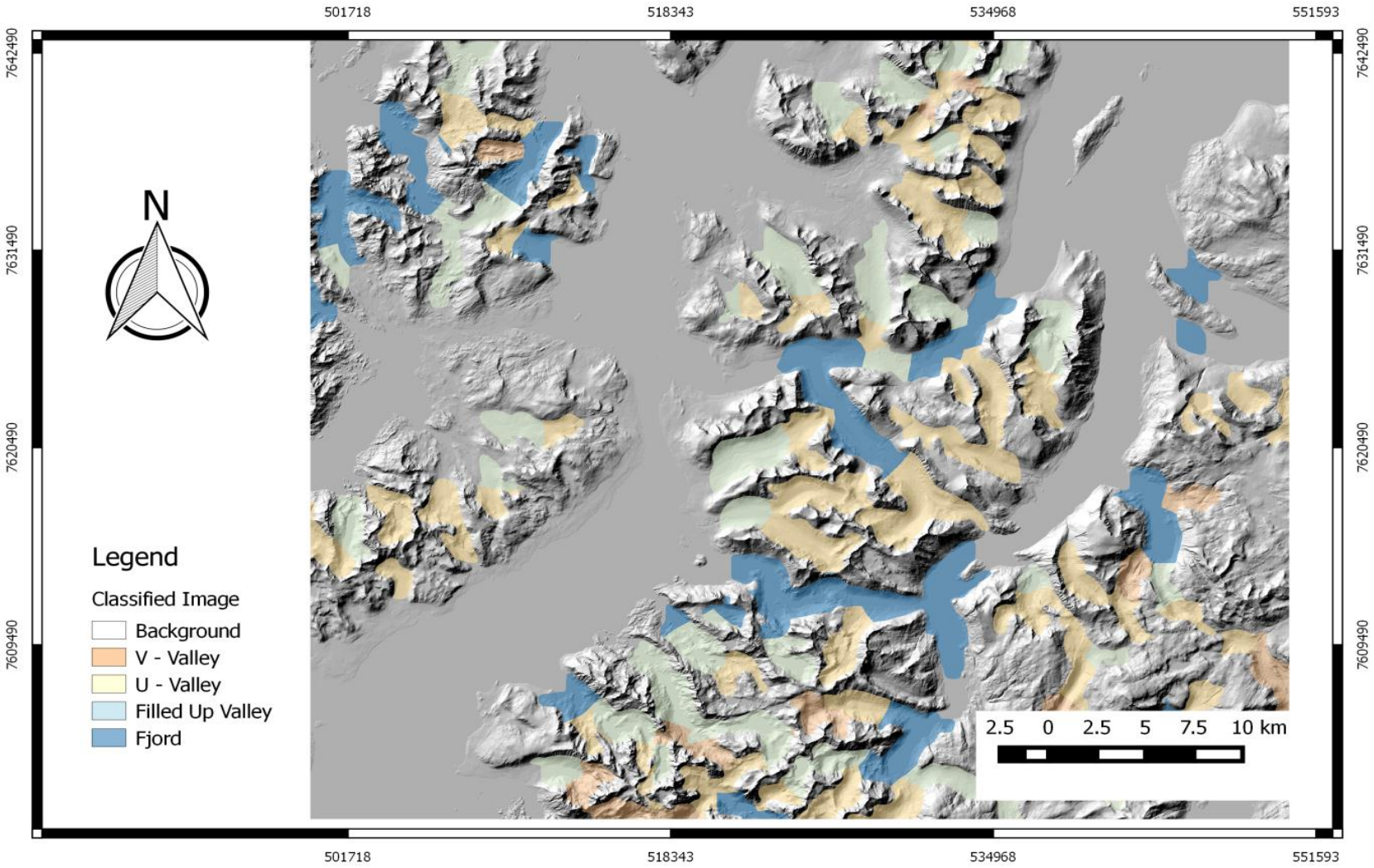


Figure 4.6 Valley detection in Norway Lofoten area. The area has a difficult topography being surrounded by fjords, but the algorithm still manages to detect u-valleys and v-valley. Location Lofoten Islands Norway

4.1.1 Evaluation Statkart 10 m

Assessing the accuracy of such a classification is challenging. The v – valley in figure 4.7shows how irregular a valley can be, by drawing three cross sections. Even though this is the same valley, classified as one object, the shape of it changes throughout its length. Judging from the first cross section the valley seems to be strong v-shaped with convex sides. The second cross section shows a wider valley bottom, characteristic to u – valleys, and the third cross section shows a u – shaped valley, the sides becoming concave.

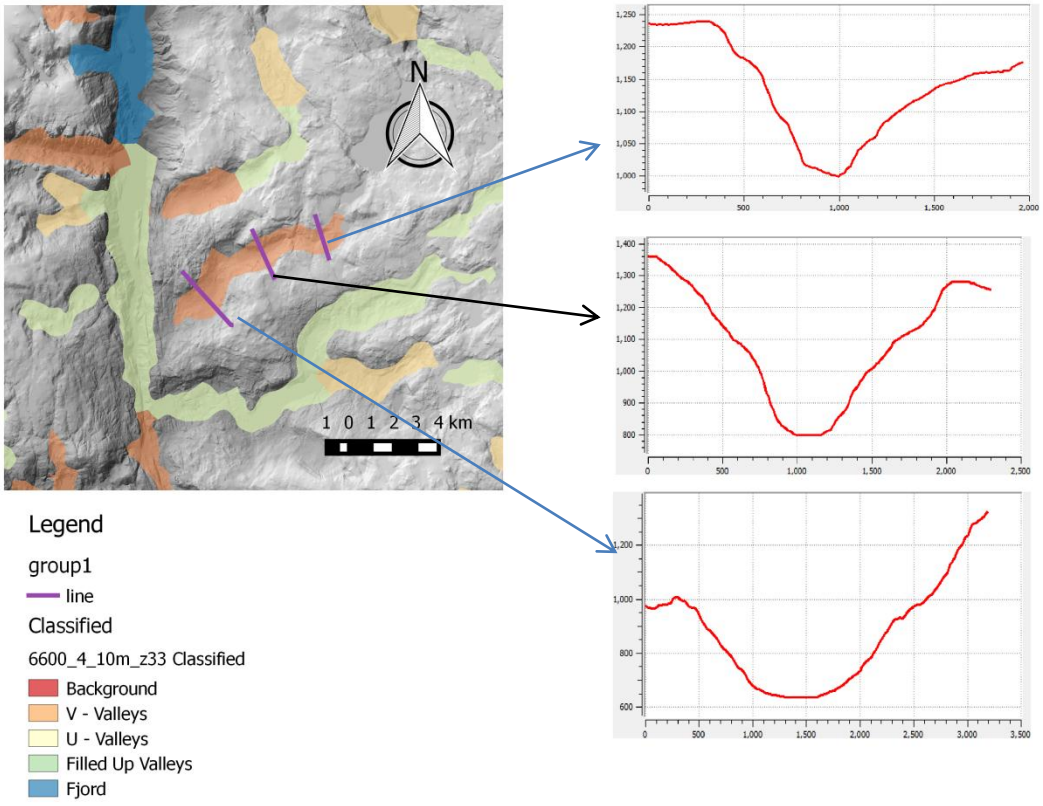


Figure 4.7 Cross section of v – shaped valley. First two cross sections show that the valley is v – shaped , whereas the third cross section shows that the valley is u shaped. This shows the difficulty in the classification and the error assessment process

Yet another issue which arises from Black – Top – Hat are small objects, which are cavities, and not exactly valleys. Figure 4.8 has three objects, classified as filled up valleys, but in essence it is debatable whether these are valleys or not.

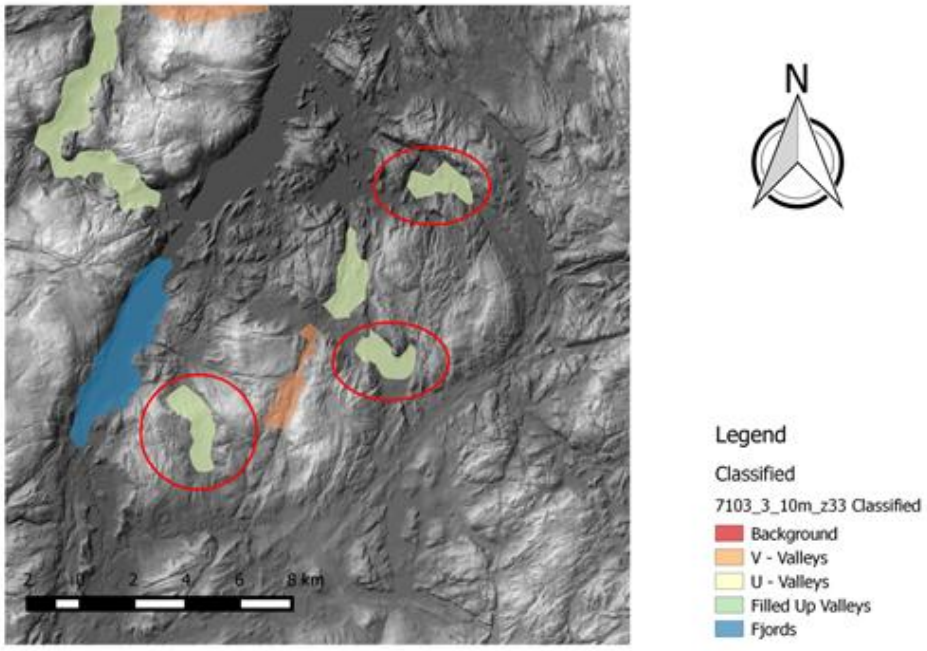


Figure 4.8 Uncertain image features classified as filled up valleys. Image objects are identified in the Black – Top – Hat transform and further classified as valleys.

In some cases, valleys are interrupted, or not totally identified as shown in figure 4.9. A filled up valley segment is identified as a small patch, instead of a whole. Fixing this issue would require a larger structuring element.

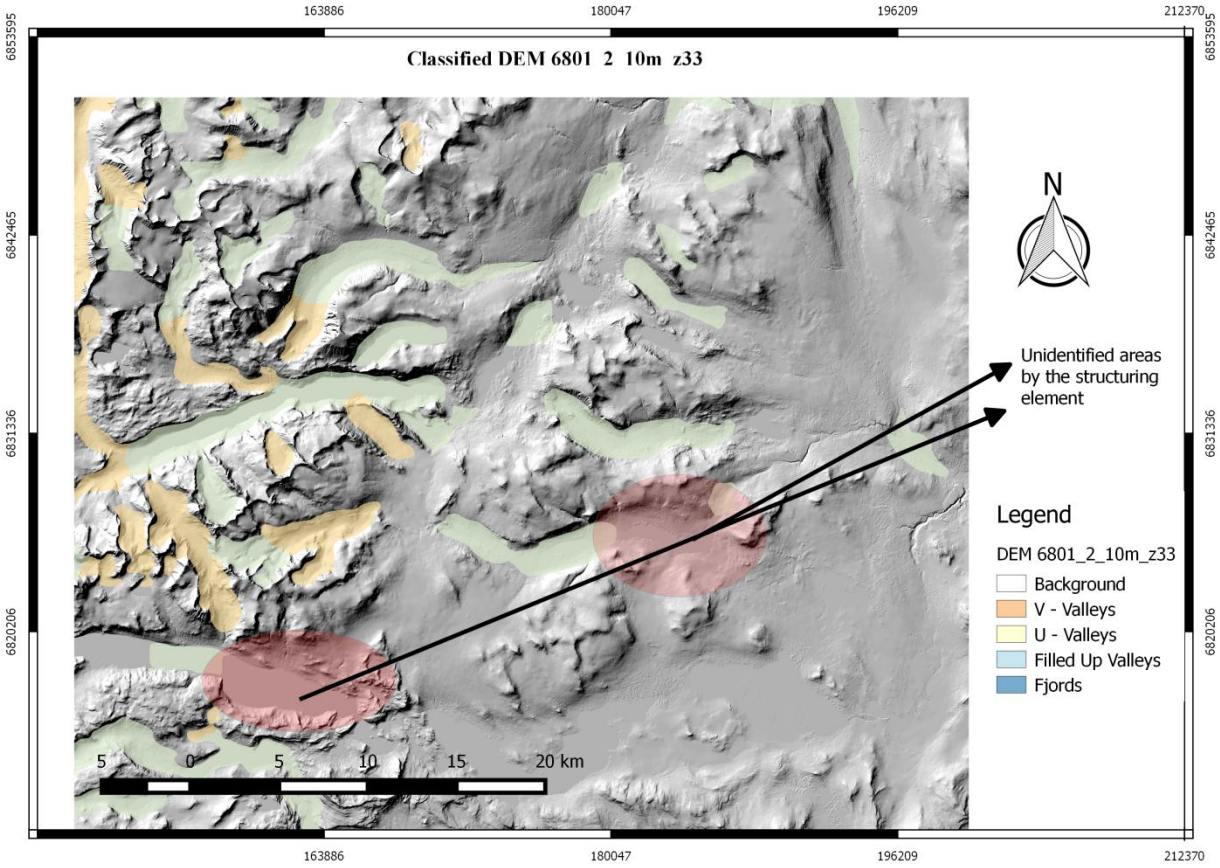


Figure 4.9 Problems that arise due to small structuring elements. The circled ellipses show areas where the structuring element did not identify the valleys. One way to solve it would be to increase the structuring element size.

In order to assess the accuracy of the classification twelve DEMs were subjected to the algorithm. The digital elevation models were carefully selected to make sure that they do not contain information that was used to train the classifier. The images were classified using the parameters described in section 3.10 and were summarized in a confusion matrix. This confusion matrix contains 30 samples of all of the valley types identified in the classification process.

Table 4.1 Confusion matrix for the Multivariate Gaussian classification. Three classes are present, v – Valleys, u – Valleys and Filled up Valleys.

	Classes	Estimated class labels		
		V – Valleys	U – Valleys	Filled Up Valleys
True class labels	V – Valleys	24	5	1
	U – Valleys	2	26	2
	Filled Up Valleys	0	0	30

According to the confusion matrix the class which is best classified is the filled up valleys. All of the 30 samples taken were filled up valleys. The second best behaving class is the u-valleys where 86% percent of the samples were classified correctly. V – valleys are the worst performing class with 79% of the classes being correctly classified, the majority of misclassified objects from these were u – valleys.

4.2 Results ASTER GDEM Norway and Romanian Carpathians

Aster GDEM was selected because it has a coarser resolution than the Statkart 10 meter grid. The hypothesis is that the method can be used on coarser digital elevation models, such as ASTER, which has a grid size of 25x25 meters. Furthermore a secondary area was chosen in the Romanian Carpathians, in order to see how different topography influences the method.

Due to a larger grid size, all of the parameters need to be adjusted. Firstly the primary Gaussian filter is reduced to $\sigma=10$. Afterwards the structuring element size is reduced to (80,80) pixels. The secondary Gaussian filter has the size of $\sigma=10$. In the skeletonization process, the threshold of the smallest admissible skeleton is reduced to 40 pixels. The area threshold is reduced by a factor of 6.25, in order to adapt the coarser grid size.

New samples were taken in order to train the classifier. The results are summarized in appendix IV. As a first impression, the hillshade in figure 4.10 of the GDEM ASTER from Norway displays high amount of noise. Without evaluating the precision of the Gaussian classifier, it can be said that the method works as expected until the kNN classification step, being able to segment the Black – Top – Hat transformed image into valley segments.

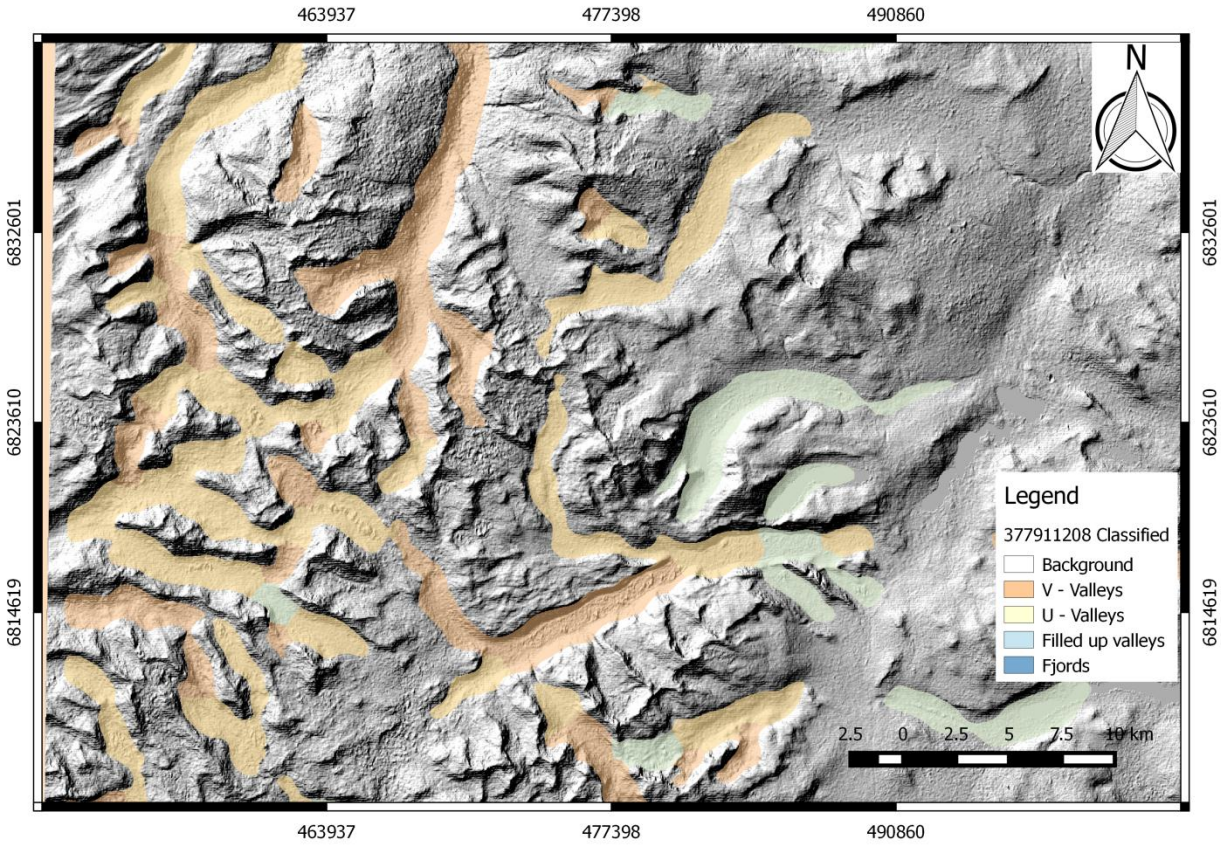


Figure 4.10 Classification of Aster GDEM in southern Norway. The results are acceptable until the kNN classification. The hillshade of the elevation model displays lots of noise.

Using the same parameters, the identical steps were applied to Aster GDEM on the Southern Carpathians in Romania. Figure 4.11 further demonstrates, that the method is independent of location, and could be used in different countries. Even though the elevation model is noisy, a visual inspection shows that the classification works well up until the Gaussian classifier. The predominant valley types in the Romanian Carpathians are v – shaped valleys.

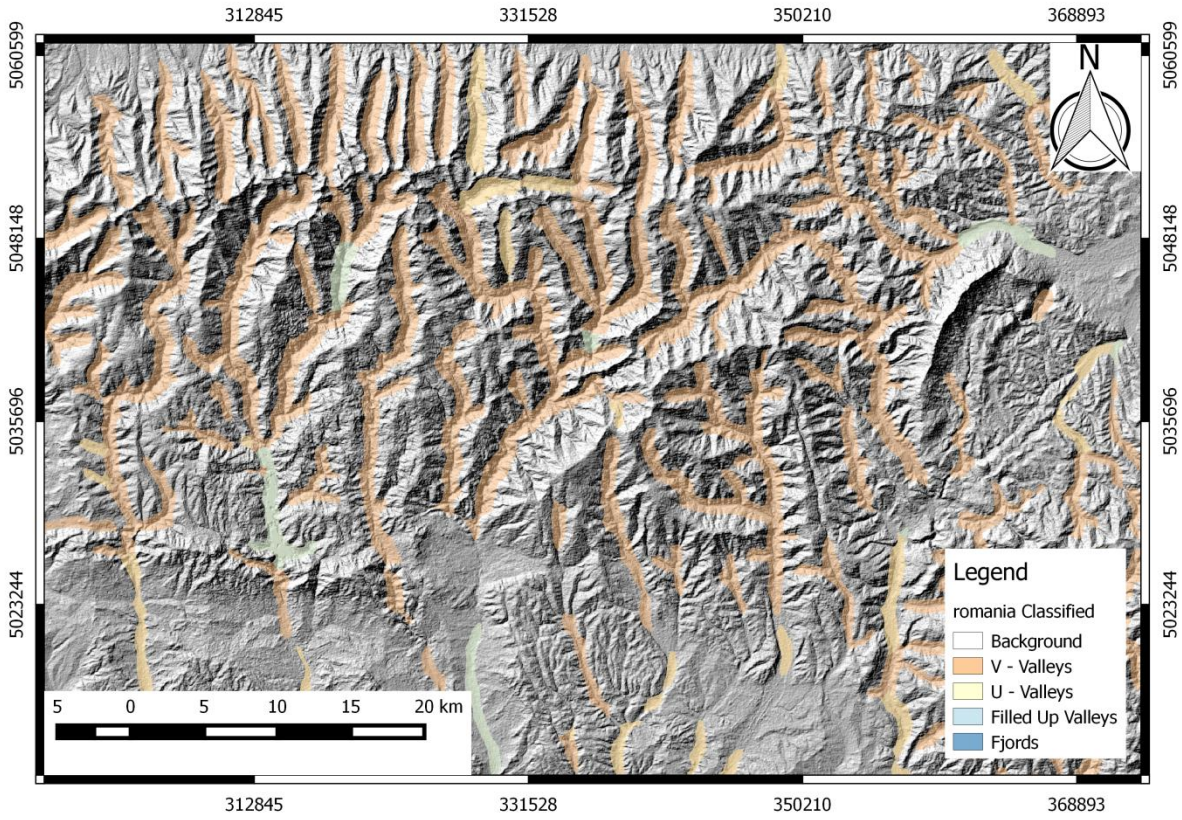


Figure 4.11 Classified Aster GDEM in the southern Carpathians in Romania, using the same parameters as the GDEM Norway. The method works well even in zones with different topography than Norway.

4.2.1 Evaluation Aster GDEM

The ASTER GDEM has a high amount of noise and voids are also present in some areas. The voids will be identified by the algorithm (figure 4.12) and will be classified as either v – valleys or filled up valleys. The amount of noise is introduced in the DEM can be observed in figure 4.13 where a cross section of a u – valley is taken from both Statkart 10 m elevation model and Aster GDEM. These rough edges have a negative impact on the classification, in the majority of cases, many u – valleys are misclassified as v - valleys. This noise adds spurious curvature and slope values which result in a faulty classification.

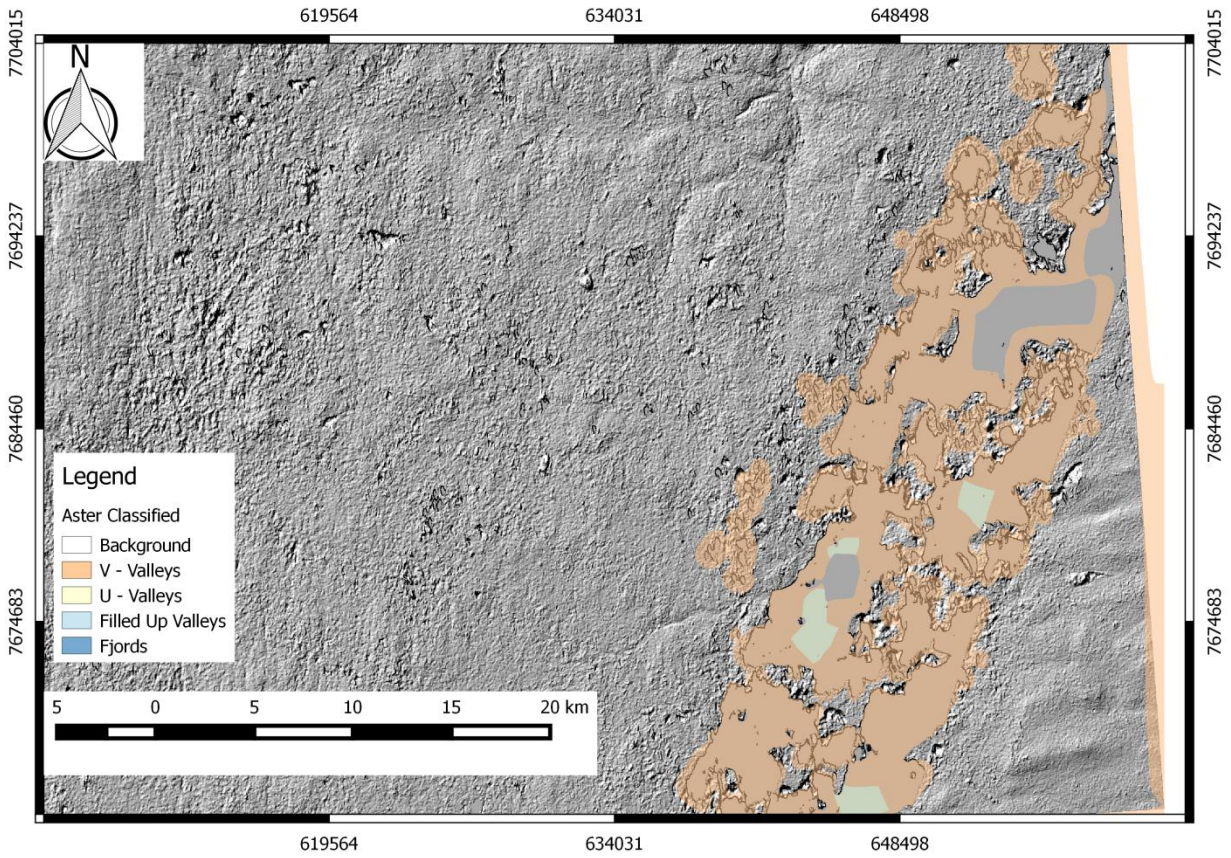


Figure 4.12 Voids in ASTER GDEM. The voids are identified by the algorithm and many times are classified as v valleys or filled up valleys.

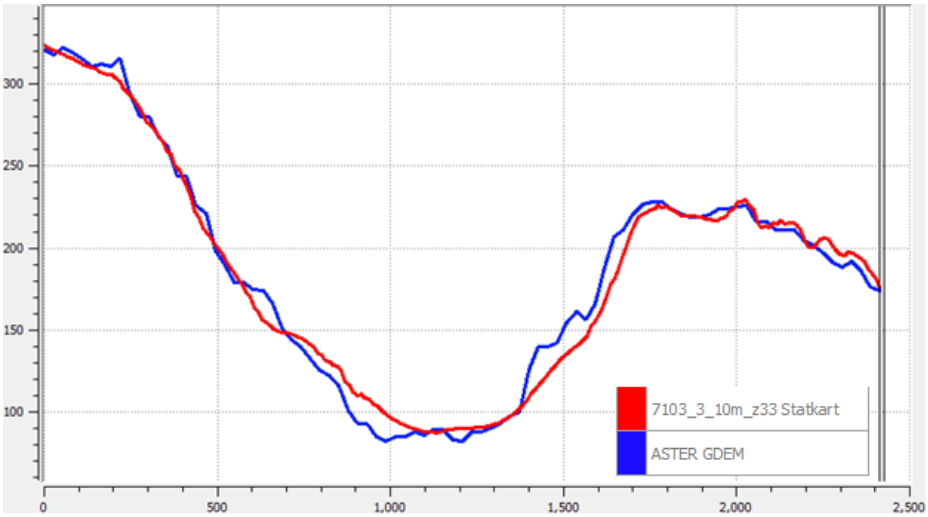


Figure 4.13 Cross section of Statkart 10 meter grid and ASTER GDM of a u – valley. The noise influences the computation of slope and curvature, and is the main source of a bad classification

Data was collected from the classified matrix in order to create a confusion matrix 15 samples were analyzed from the 3 types of values and the results were summarized in table 4.2:

Table 4.2 Confusion matrix for ASTER GDEM Norway. True class labels are the actual labels that should be assigned, estimated class labels are the labels that the Gaussian classification assigned

True class labels	Classes	Estimated class labels		
		V – Valleys	U – Valleys	Filled Up Valleys
	V – Valleys	6	7	2
	U – Valleys	1	11	3
	Filled Up Valleys	1	1	13

Poor results can be observed for the v – valleys, most of them being classified as u valleys. Out of the 15 u – valley samples 73% are actually u valleys, and 87% of the objects classified as filled up valleys are actually filled up valleys. Many u – valleys can be filled up valleys, and are misclassified often due to the noise. In order to generate the ASTER elevation model image matching is used. The filled up valleys are usually occupied by water, and even though ASTER GDEM V.2 is corrected for water bodies (Tachikawa et al., 2011) spurious matches can still be found on water, this causes difficulty to differentiate between u – valleys and filled up valleys.

4.3 Results Statkart 50m DEM

NOTE: The method uses the Gaussian Classifier trained with Statkart 10m DEM, because acquiring the samples necessary to train the classifier extends beyond the time limit of the thesis.

In order to subject the 50 meter resolution grid to the algorithm, changes need to be applied to the parameters. The primary sigma filter was reduced to $\sigma=5$. The size of the structuring element was set to (40,40), and the secondary sigma filter was also reduced to $\sigma=5$. Furthermore the threshold for the smallest skeleton was modified to 20 pixels and the threshold for the area was changed to 800 pixels.

The next step would be to collect new samples for the Gaussian classification, but because the elevation model is not noisy, the same parameters were used from the Statkart 10 m grid. The classification benefits from the fact that the features used to train the classifier are

normalized, making them scale invariant. Figure 4.14 depicts that classification works well, even on a coarser digital elevation model.

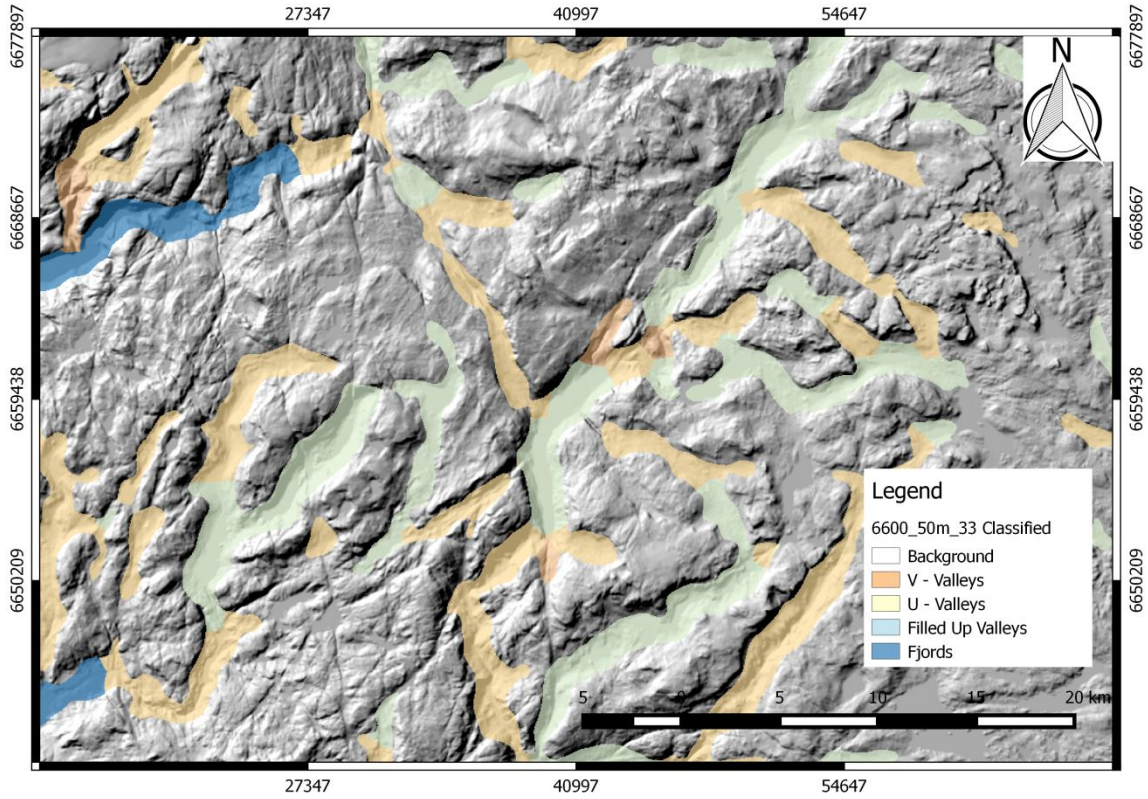


Figure 4.14 Results of the Gaussian classification on the Statkart 50m resolution grid zone southern Norway. Even though the elevation model is coarse, the algorithm still manages to classify the Black – Top - Hat transformed image

4.3.1 Evaluation Statkart 50 m DEM

The differences between the two elevation models are not major(figure 4.15). The overall shape of the valleys is maintained, whilst small variations are removed. Increasing the grid size has a filtering effect. It would be incorrect to create a numerical assessment of the results, as long as the Gaussian classifier is not trained on the 50 meter resolution DEM, however a visual assessment can be made of the results. Overall, the results are similar, but the differences arise from two main factors:

- The 50 meter grid is coarser, resulting in a much smoother Black – Top - Hat transformed image, and that leads to smoother boundaries in the thresholded image. Smoother

boundaries result in less segmentation in the skeletonization process, therefore different objects in the kNN classified image. In essence the Statkart 10 m grid will have more valley segments (over segmentation) than the Statkart 50 m grid.

- Evidently the training of the Gaussian classifier has also a considerable effect. Figure 4.15 displays that the extra detail in the Statkart 10 m resolution grid will introduce various curvature and slope values, which will average out differently than in the Statkart 50 m grid, hence the necessity of training the classifier with the proper DEM.

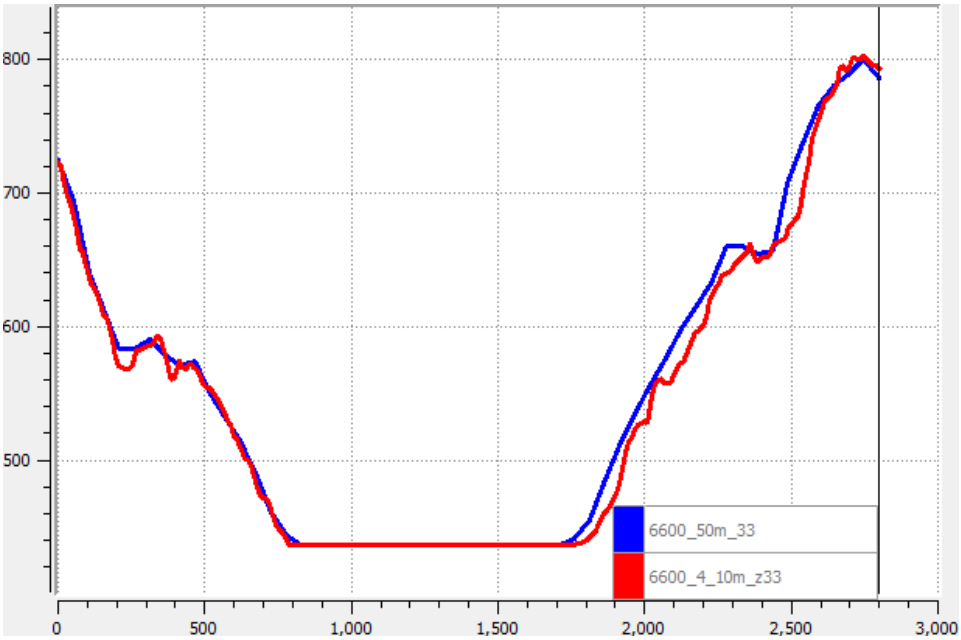


Figure 4.15 Cross section of a filled up valley. Blue line represents the cross section on Statkart 50 meter grid, the red line represents the cross section on the Statkart 10 meter resolution grid. The overall shape of the valley is not changed, whilst the coarser resolution Statkart 50 meter grid has the small variations filtered out.

Figure 4.16 was introduced in order to see differences between how the classification behaves on the two elevation models. The filled up valleys are almost entirely correctly classified, whereas differences arise in the v valley classification. The ellipses drawn on the figure highlight some of the differences between the two classified images.

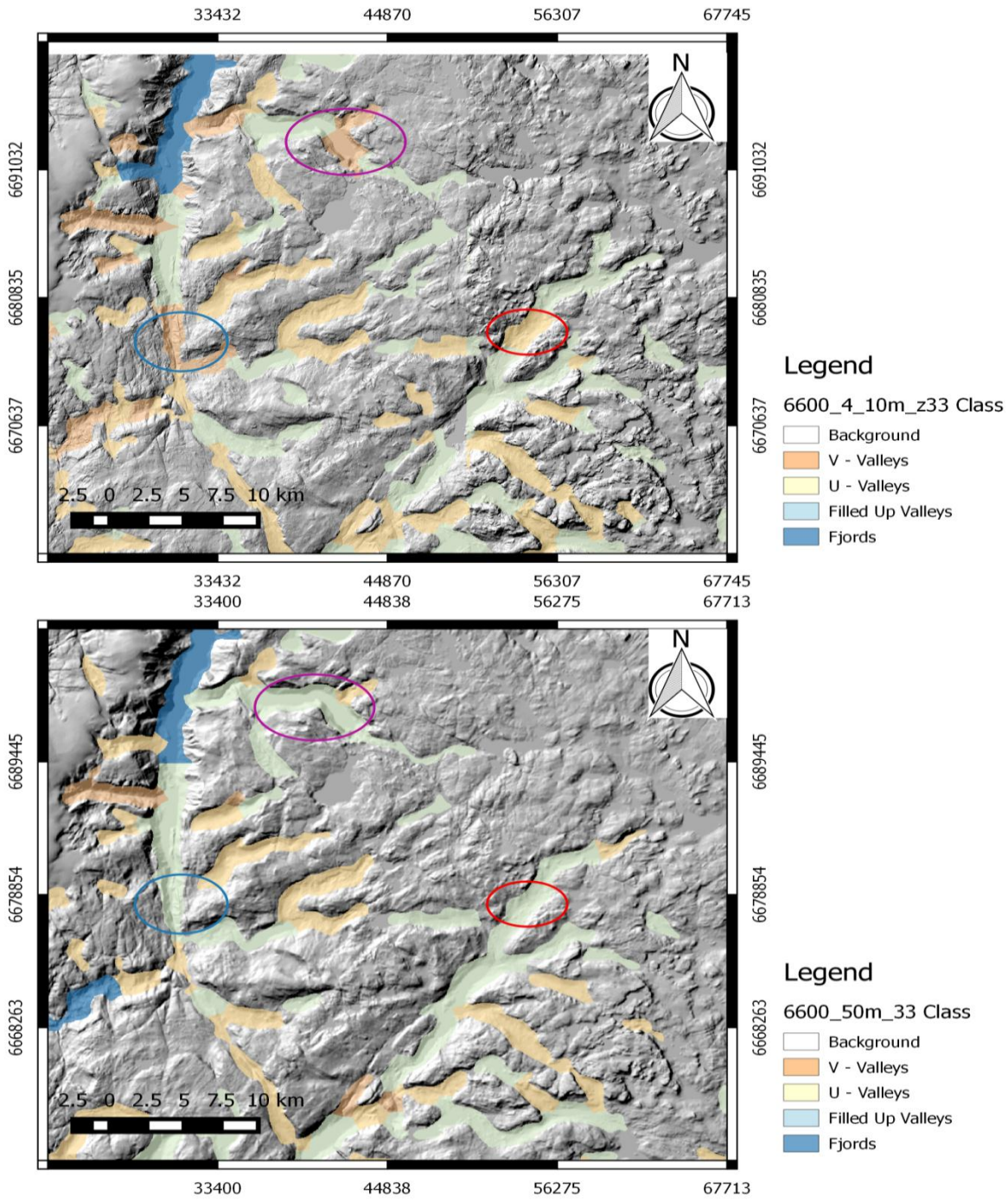


Figure 4.16 Differences in classification between Statkart 50 meter grid and Statkart 10 meter grid. The red and blue ellipses show classification differences due to over – segmentation in the Statkart 10 meter grid. The violet ellipse displays classification difference due to the Gaussian classifier.

In figure 4.16 the red ellipses represent the differences in classification due to over – segmentation of the skeletonization process. In the Statkart 10 meter model, due to higher amount of detail, over – segmentation occurs consequently an extra valley segment is created, whereas in the Statkart 50 meter model this remains one object, resulting in the whole object being classified as a single filled up valley. This difference would be present even if the Gaussian classification would be properly trained for the 50 meter elevation model.

The violet ellipse in the Statkart 10 meter grid, is a poorly classified valley segment, in contrast to the Statkart 50 meter where this is attributed into the correct class.

The blue ellipse on the 10 meter grid is attributed to v – valleys opposed to the 50 meter grid where it is classified as a filled up valley. This is largely because of over – segmentation, but in this case, the over – segmentation favors the 10 meter grid.

One can say that, the results would be much more promising if the Gaussian classifier would be properly trained, and the reduced detail from the Statkart 50 meter grid would actually have a positive impact on the final result

4.4 Results SRTM

NOTE: The method is presented until the KNN classification step. The Multivariate Gaussian Classification is not applied, because acquiring the samples necessary to train the classifier extends beyond the time limit of the thesis.

The SRTM elevation model was introduced to this study in order to test the limits of the method, this being the coarsest elevation model of all of the above. The settings that needed to be applied were the following: primary sigma filtering was $\sigma=3$. The structuring element was set to (35,35) and the secondary sigma filter was modified to $\sigma=2$. The threshold for the smallest skeleton was changed to 15 and for the area the value was altered to 2800 pixels. The algorithm was completed until the KNN classification step. The results can be seen in figure 4.17. The figure shows that even in such condition the algorithm works. The next step would be to gather the necessary information to train the Gaussian classifier.

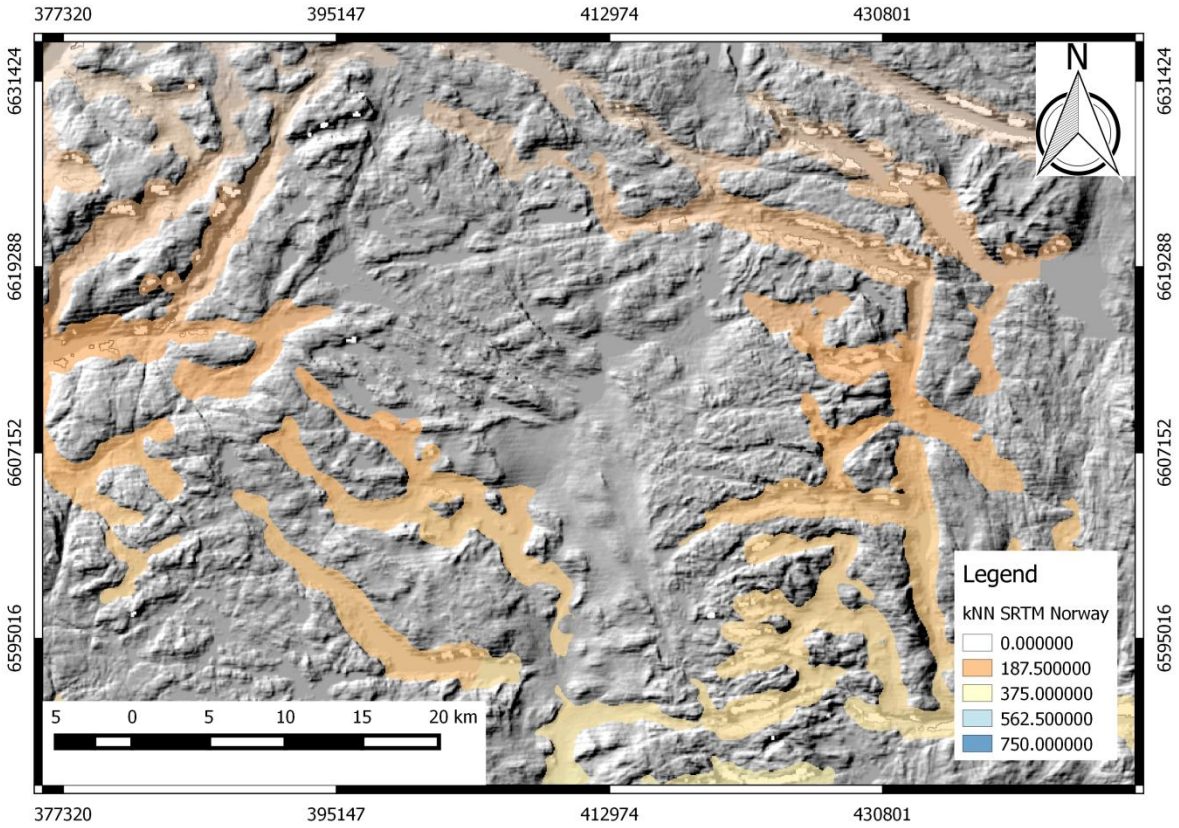


Figure 4.17 kNN classified SRTM elevation model in southern Norway. The Black – Top– Hat transformed image can be used to detect valley segments, even with such a coarse resolution

The valley segmentation and classification processes are separate from one another, therefore the collected samples that need to be used to train the Gaussian classifier influence directly the outcome.

It is worth mentioning that the greatest drawbacks of the SRTM elevation models are the voids. The majority of them are present on the valley sides, considerably reducing the amount of information that could be used to train the classifier.

If the training set accounts for the voids, and a good training set is collected, there is no reason why the valley classification step wouldn't work.

5. Discussions

5.1 On the filtering

5.1.1 Filtering for the Black – Top – Hat

Figure 5.0 and figure 5.1 show the results of a mean and a median filtering applied to a valley section within a DEM. The main reason of this initial filter is to reduce local variation within a valley. This smoothens the DEM and filters shallow cavities. A mean filter and a median filter, in this case used on a15x15 window does not have the desired effect to smoothen out details. The sigma filter on the other hand removes all of the small variations from the valley sides, distorting it into nearly a smooth wave, in order to create adequate conditions for the Black – Top – Transformed image.

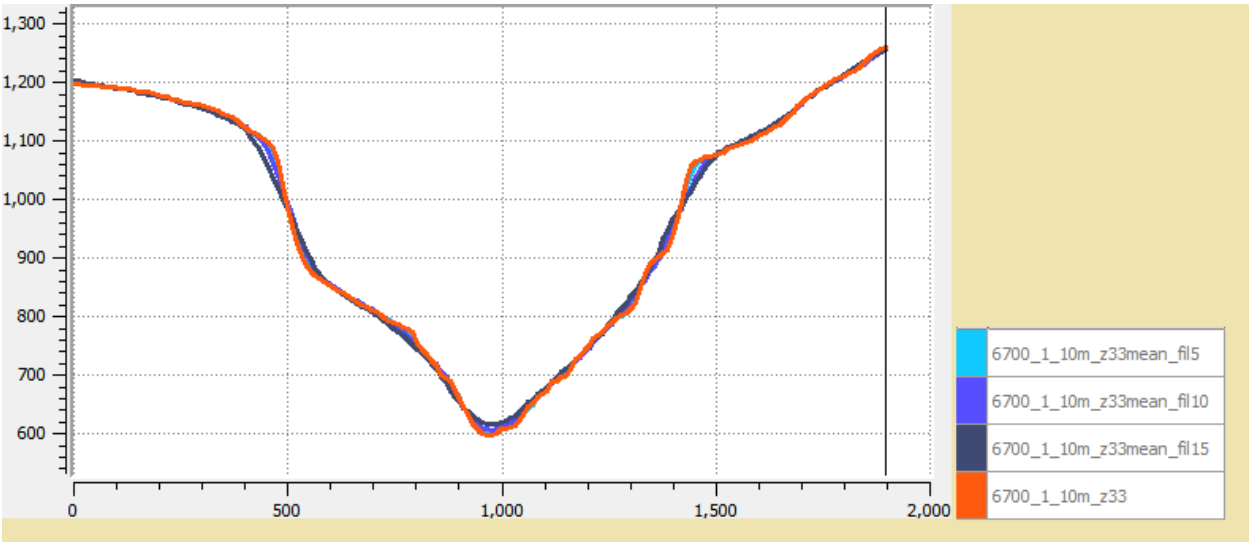


Figure 5.0 Mean filter applied to a valley segment. The size of the mean filter is changed gradually from 5x5 (light blue) to 10x10 (dark blue) to 15x15 (dark gray). The compared to the original DEM (orange) small variations are eliminated, but in essence the form of the valley does not change.

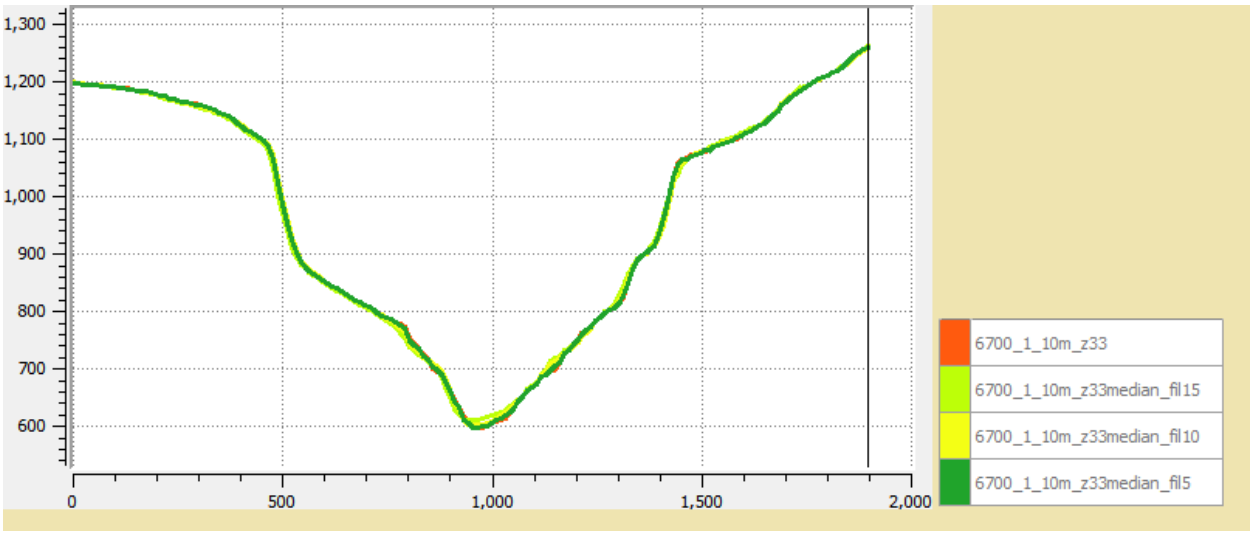


Figure 5.1 Median filter applied to the elevation model at different window sizes. Orange denotes the original DEM, dark green illustrates the effect of a 5x5 median filter, yellow represents the result after a 10x10 median filter and bright green illustrates a 15x15 median filter. The results look similar to each other, the minor variations are filtered out, but in essence the form of the valley does not change.

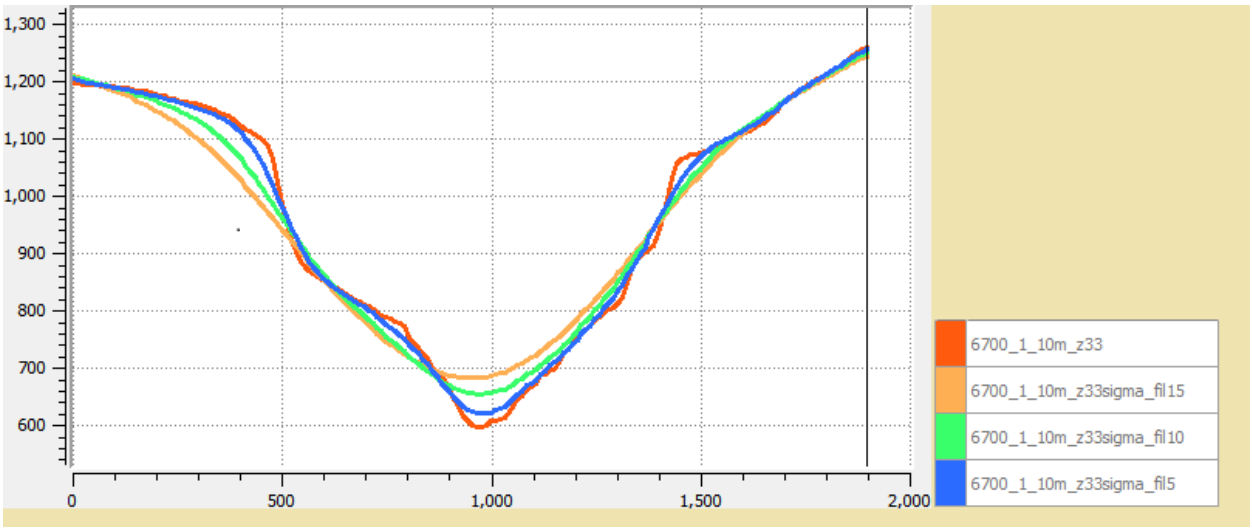


Figure 5.2 Sigma filter on the digital elevation model. Red represents the original DEM, whereas blue is the sigma filtered DEM with σ=5, green has σ=10 and orange is the result of the DEM subjected to a σ=15. On the large sigma filter the essential form of the valley changes also. Major changes can be seen on the bottom of the valley and the sides. This has a positive effect in the Black – Top – Hat algorithm

Skeletons are highly sensitive to boundary changes. Applying a filter which is not strong enough will result in a higher variation in the boundary of the Black – Top – Hat transformed image, creating a higher number of objects. Figure 5.3.A illustrates that a mean filtered image has a

higher number of valley segments than the sigma filtered image (Figure 5.3.B). Reducing the detail in the DEMs becomes a necessity in order to avoid over-segmentation of the Black – Top – Hat transformed image.

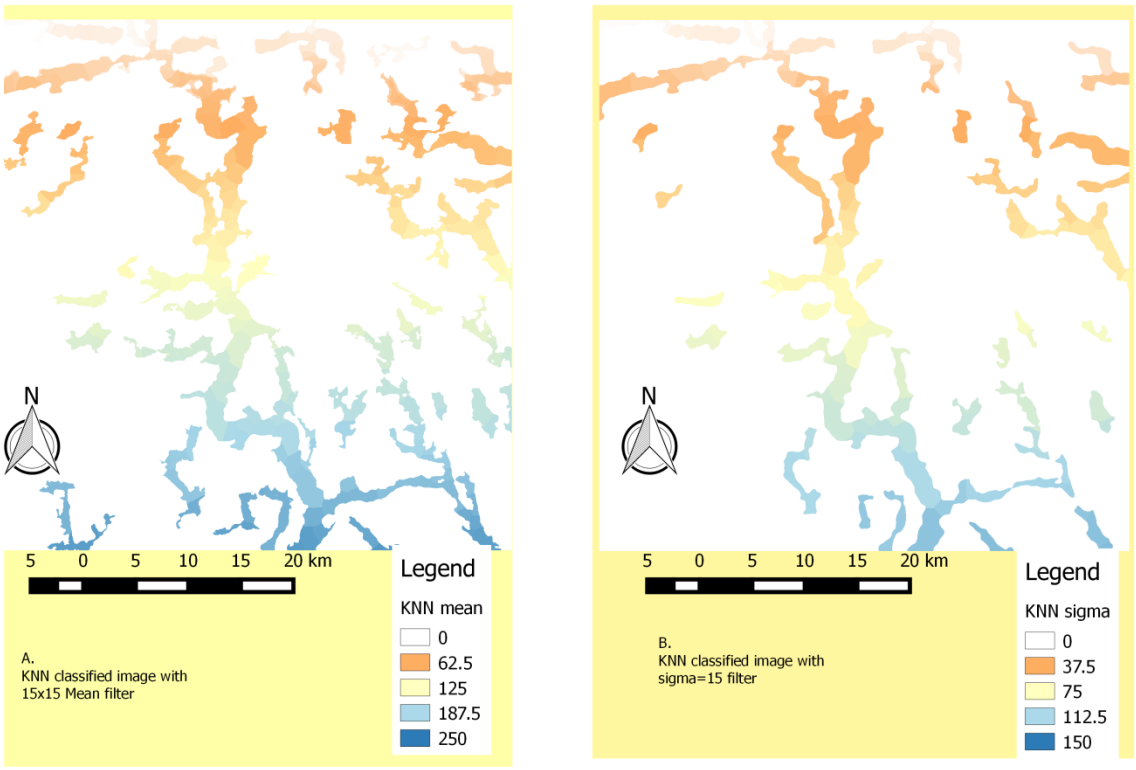


Figure 5.3 Top hat transformed image with 15x15 mean filtering (left), Top hat transformed image with σ=15 Gaussian filter. The mean filtered image does not smooth the edge of the image objects enough, resulting in a higher segmentation compared to the sigma filtered image.

5.1.2 Filtering for Slope and Curvature

An elevation model such as Statkart 10 m grid exhibits high amount of variation this is smoothed out with a σ=1 Gaussian filter. Figure 5.4 presents the mean curvatures of a valley segment, on the raw elevation model, and on the sigma filtered elevation model. Removing these local variations improve the slope and curvature values. As illustrated on the right segment of figure 5.4, concave and convex surfaces tend to be joined together. The method would work

without this filtering step, but the local variations do have a negative impact on the classification step.

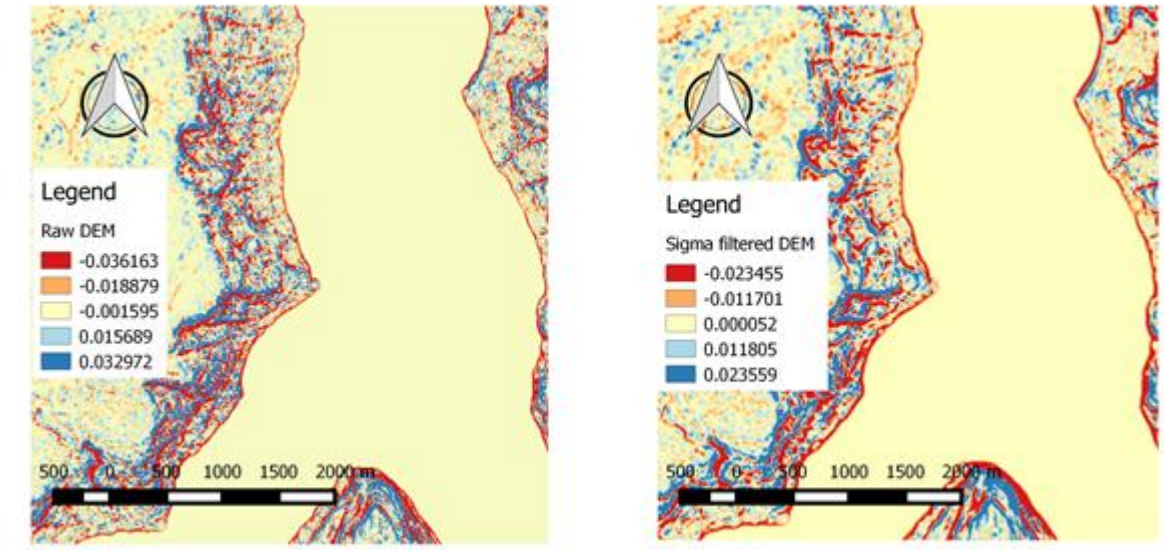


Figure 5.4 Left image illustrates the curvature of the raw DEM, right image represents the curvature of the surface after a small σ=1 Gaussian filter. This shows that local small variations are elliminated and the surface is smoothed out. The convex and concave objects are joined togheter, instead of being distributed in a "salt and pepper" pattern.

5.2 Thresholding

Figure 5.5 A exhibits a Black – Top – Hat transformed image. The image demonstrates in great detail how valleys and cavities appear on the Earth's surface. In order to go proceed to the skeletonization process, the foreground needs to be segmented from the background. If all negative values would be admitted to segment the image, then Figure 5.4 B would be the result. This cannot be used further in the skeletonization process. The high amount of details needs to be removed in order to enhance only the deep valleys. Finding a threshold value is also not a trivial matter. In case of Norway, the -100 value was selected on a trial and error basis. This value is entirely dependent on topography, while flat zones require values closer to 0. Mountainous regions require larger negative values in order to achieve a good segmentation.

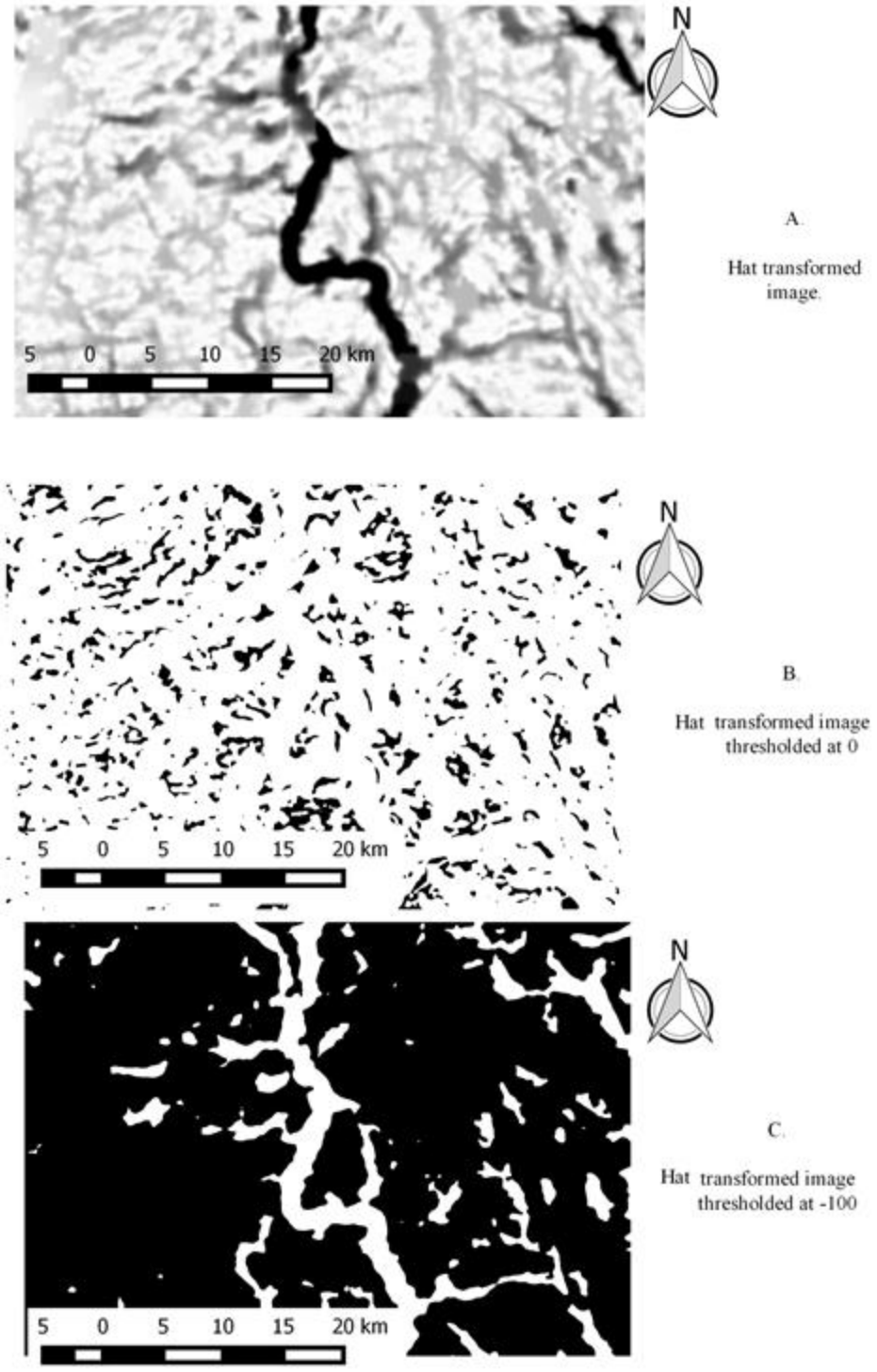


Figure 5.5 A. Black – Top Hat transformed image, B – Transformed image thresholded at 0. C – Transformed image thresholded at -100 meters. If no additional threshold would exist the image detail would overwhelm the skeletonization algorithm.

5.3 Flow accumulation versus image skeletons

It is debatable whether image skeletons or flow routing algorithm should have been used in the KNN classification. The flow routing algorithm has the advantage that it takes into account slope and accumulation. Figure 5.6a,b shows exactly why flow accumulation would be beneficial. The Black-Top-Hat identifies cavities, which can be uncertain. Adding flows to the KNN, would segment these elements according to where the stream is. The image skeletons take into account only the variations on the side of the object, therefore every noise will create a new skeleton, achieving unnecessary segmentation. Another issue that is presented in the figure is valley segments which should be separated. The figure shows a filled up valley which is essentially classified as one single valley. In reality there is a peak in the middle of the valley. If a flow routing algorithm would be used in the kNN process as the training image, there would be two separate flows towards east and west.

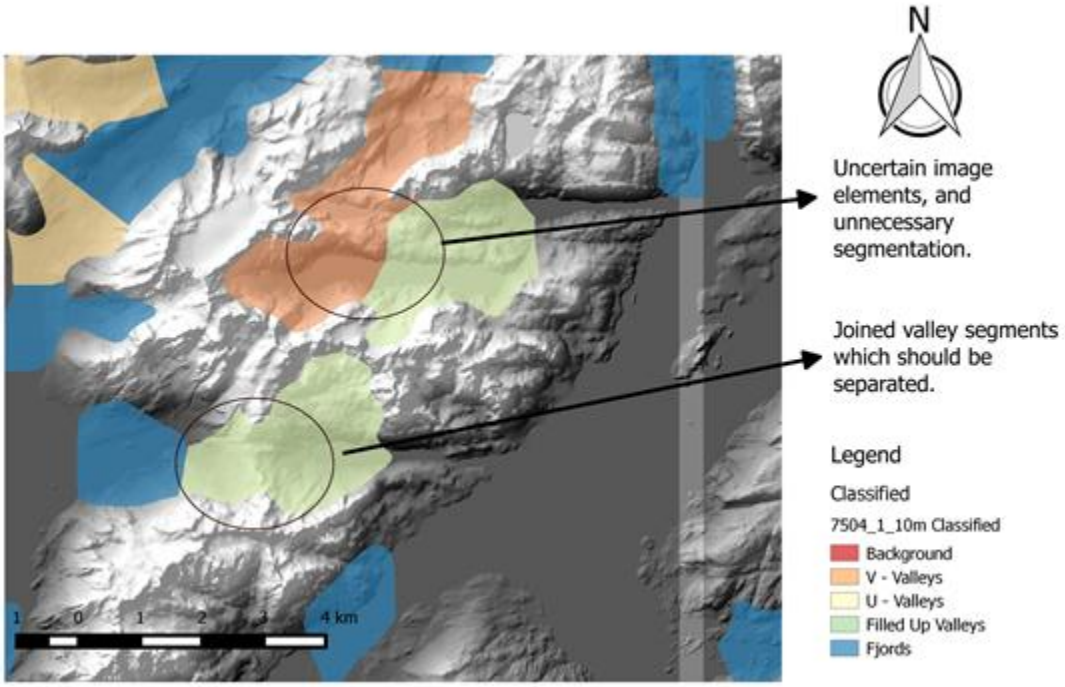


Figure 5.6.a Uncertain image elements, and unnecessary segmentation is a shortcoming of using Black – Top – Hat and image skeletons. By using flow paths the over segmentation problem could be possibly avoided. The same would also separate the joined valley segments, classifying them in two different objects

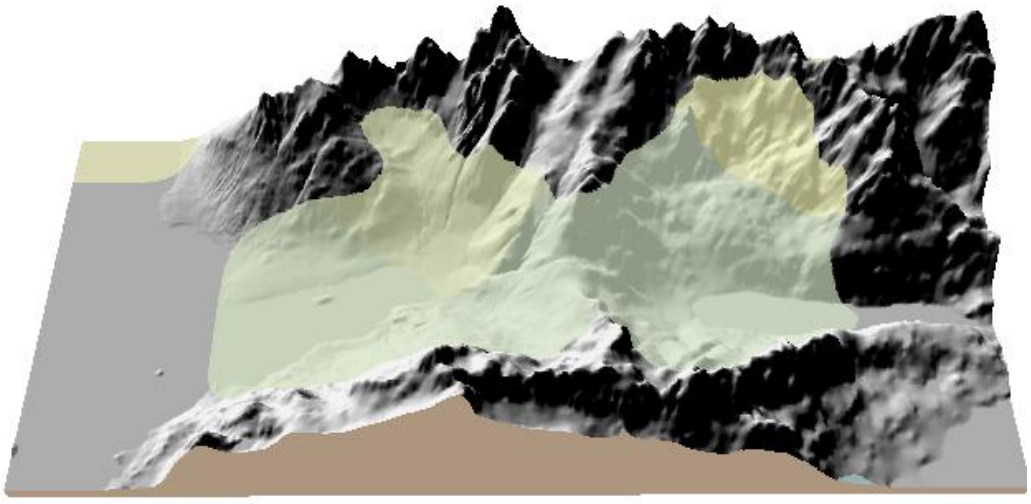


Figure 5.6.b 3-D view of joined valley segments. The top – hat transform and the kNN classifies this as one object.

Another good example of joined valley segments can be observed in figure 5.7 where this peak is more subtle. The object is classified as a U valley, and a longitudinal profile was created in order to exemplify the above mentioned problem. The longitudinal profile shows a strong peak a quarter along the way, showing that in theory these objects should not be joined together and they should be treated as different valley segments.

Skeletons and flow accumulation are not two disjoint elements. These are in a relationship, and in many cases these can be observed close to each other, as they both are part of the same landform. Figure 5.8 depicts both skeletons and flow accumulated areas, and shows how well one follows another, with minor differences.

The main reason why skeletons were chosen over flow routing is observable in figure 5.9. Even though one can eliminate the local pits, it is still highly probable that the flows will not converge to the global minimum within a digital elevation model. Usually local flats can be considered as minimums by algorithms as D8, and the flow routing will stop. This is the case in the majority of times for the filled up valleys. Having valleys filled up with water will essentially stop the flows.

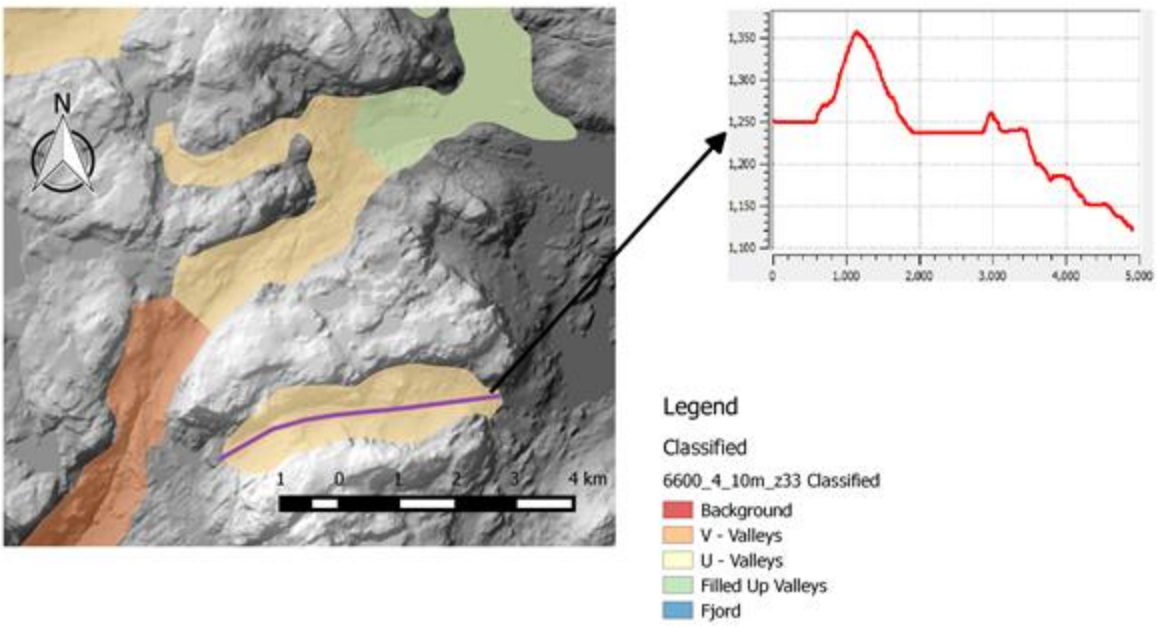


Fig 5.7 Longitudinal profile of image object which is classified as U valley. Even though the object is identified as a valley, on the length of the longitudinal profile a peak can be seen. This could be easily eliminated if instead of skeletons, flow accumulation would be used in the KNN classification step

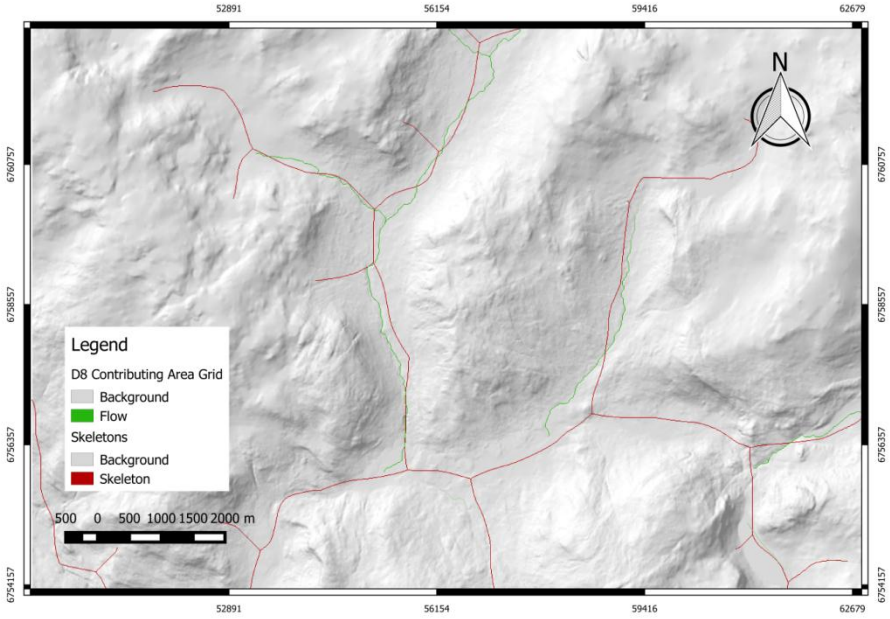


Fig 5.8 Flow paths (green) versus image skeletons (red). Even though the two are created in a separate way, these are always in close proximity, as they are part of the same object

The skeleton splitting algorithm attains good outcome in splitting, but also a skeleton pruning algorithm could be employed for eliminating the small skeleton segments.

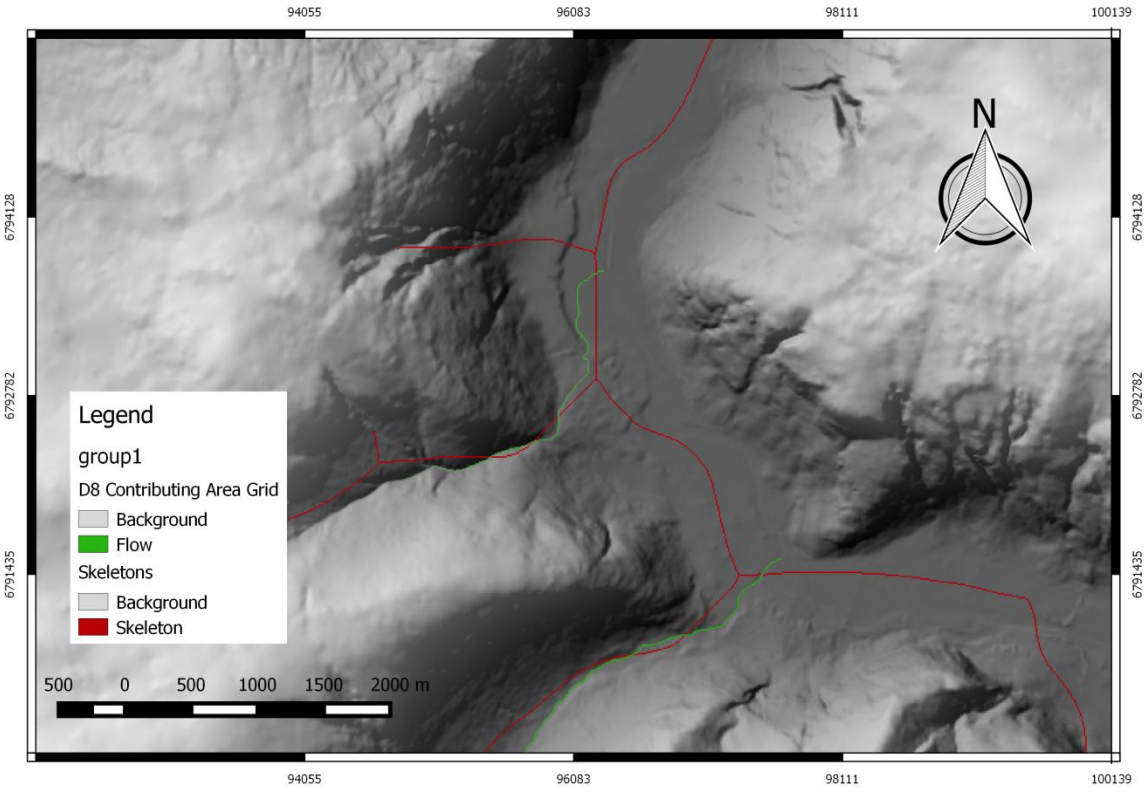


Fig 5.9 Shortcomings of the Flow accumulation areas versus the image skeletons. Flow paths (green) lack continuity in case of filled up valleys, which would result in foul classification in the KNN classification step. Image skeletons (in red) are always connected to each other, eliminating the before mentioned problem

5.4 Scaling

In Chapter 3 it was indicated that the sizing the structuring element is a sensitive issue. When a large structuring element is used then large valleys can be identified. A small structuring element can be used to search for ravenes on digital elevation models. By setting a smaller threshold of -10 meters and a structuring element of 21x21, ravenes and ridges can be found on the digital

elevation model. To classify them, further samples would be necessary, but the methods used in mathematical morphology show in figure 5.10 how small objects can be also identified.

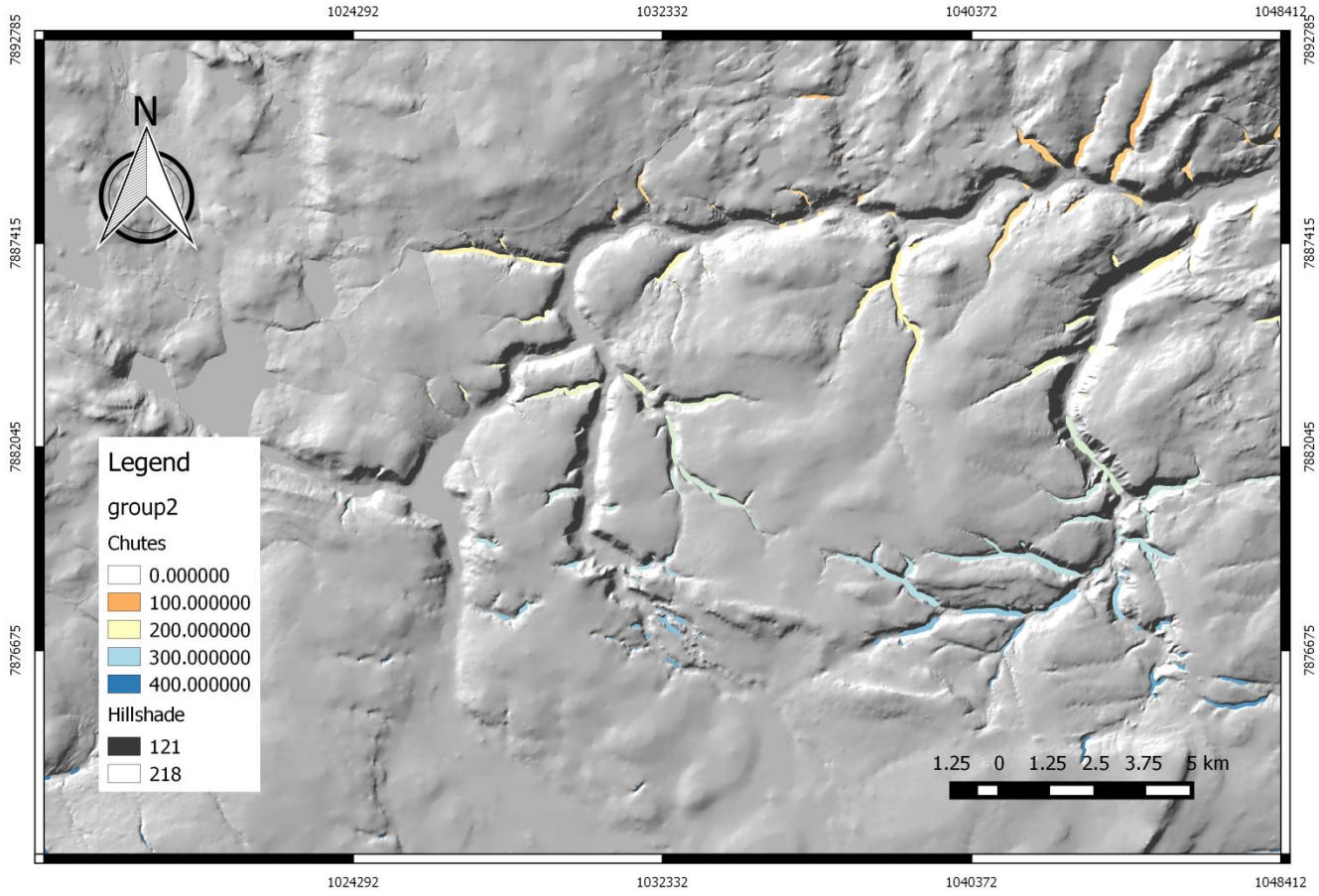


Figure 5.10 Chutes/ ravines using the Black – Top – Hat transform. The bottom of the larger valleys are ignored because of the small structuring element. In this case a 21x21 pixel structuring element was used, with a minimum threshold of 10 meters. Many of the objects identified could be classified, given a good training sample.

5.5 Relation to other work.

Landform classification has been successfully executed in the past using fuzzy logic and heuristic rules by MacMillan et al. (2000), who successfully classified landforms into 15 landform elements using 10 features. By re-grouping the 15 landform elements into four major elements MacMillan et al. (2000) managed to separate upper, mid, lower and depressions. The algorithm managed to identify valley bottoms from the combination of the landform elements (

figure 5.11). The method does not approach the valley form classification, nor does it approach the issue of scalability, additionally dependencies between landforms need to be explained.

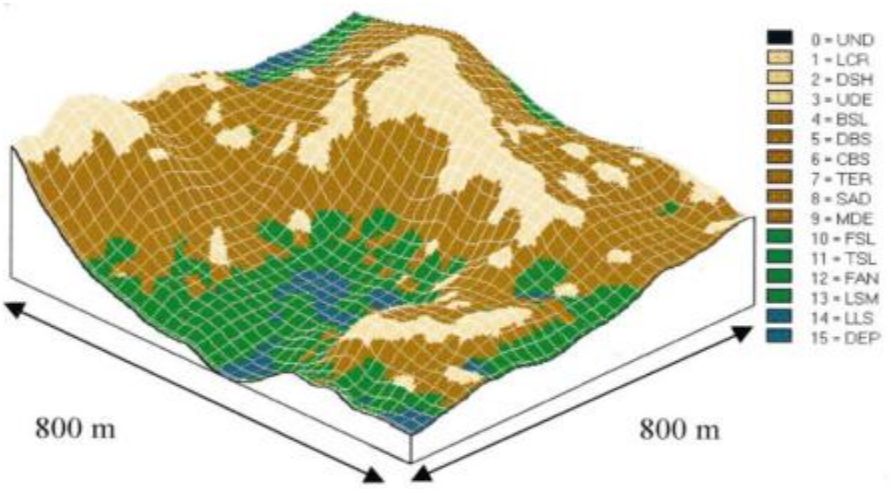


Figure 5.11 Simplified 4 unit landform classification after MacMillan et al. (2000)

Schmidt and Hewitt (2004) used fuzzy classification and added gray scale morphology to aid the classification steps. The authors first executed a fuzzy classification of slope and curvature (form element classification), afterwards a Top – Hat fuzzy classification (landscape position indexes) and a final fuzzy classification for the landform elements. By combining the form classification and their terrain classification they managed to produce a "Valley index" map presented in figure 5.12 The authors do take into account scaling, but do not further classify the valleys.

Figure 5.12Dark areas relate to high membership values for hills/valleys; bright areas indicate low membership values. Modified after Schmidt and Hewitt (2004)

A different approach was used for landform detection by Drăguţ and Blaschke (2006) (figure 5.13),using an object based image analysis software developed by Baatz and Schäpe (2000) to delineate homogeneous landforms using elevation, profile curvature, plan curvature, and slope gradient. Afterwards the objects are classified using a relative classification model, built on the surface shape and altitudinal position of the objects. This method does not proceed to classify valleys.

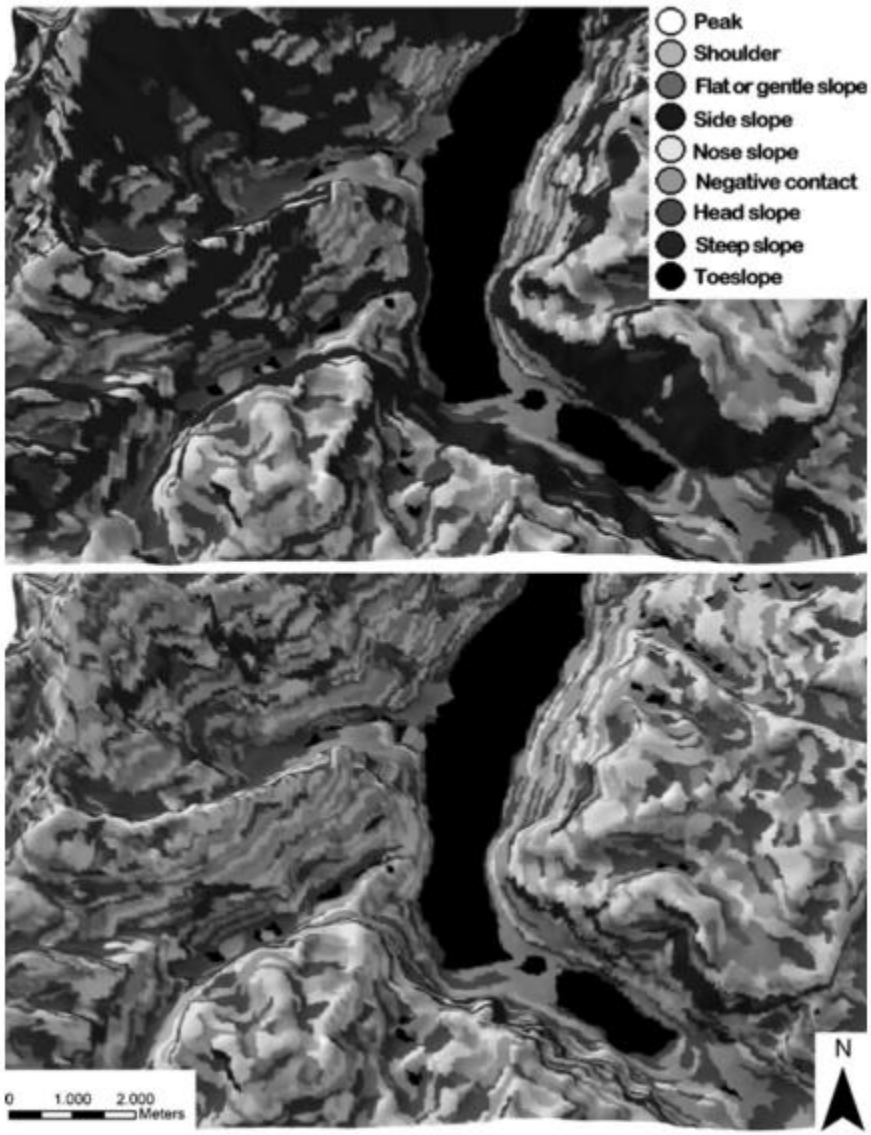


Figure 5.13 3-D visualization of landform classification of the Berchtesga-den area. Steep slopes defined by a slope gradient higher than 45° (top)and higher than 60° (bottom). After Drăguţ and Blaschke (2006)

Black – Top – Hat transforms have been used for valley detection first by Rodriguez et al. (2002). Luo et al. (2015) used the idea to further develop a progressive black – top – hat which adapts the window size, by taking into account a slope factor to extract valleys of different orders, in order to estimate valley volumes on Mars. The method can extract valleys with high precision as figure 5.14 demonstrates.

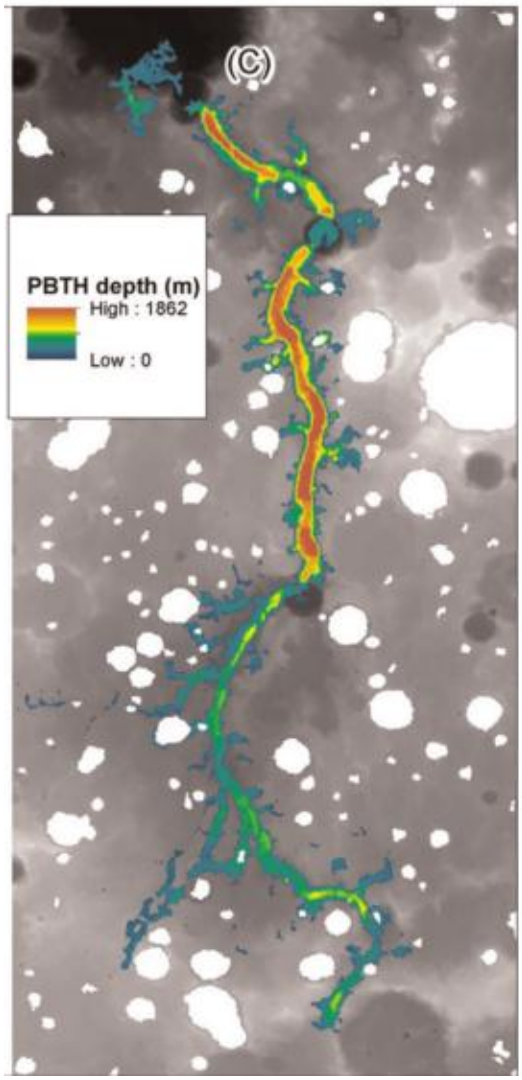


Figure 5.14 valley depth extracted with PBTH after Luo et al. (2015)

All of the above methods have one thing in common: they manage to identify valleys in their own different way, but none of them classify valley forms, it is possible to perceive that extensions of these methods which identify valley forms.

6. Conclusions and outlook

A method was proposed to identify and classify v-valleys, u-valleys and filled up valleys using Black – Top hat transform, image skeletons, kNN classification and Multivariate Gaussian classification. The method can be employed at different scales, and is not influenced by the alternating orientation of the valleys. The data for the Gaussian classifier is normalized, therefore it can be used at different scales

The methodology was conducted on the 10 and 50 meter elevation model from the Norwegian Mapping Authority, ASTER GDEM in two locations in southern Norway and in the southern Carpathians in Romania, and SRTM 3 arcsec elevation model. The main relevant findings can be summarized as follows:

- Satisfactory results were obtained on the 10 meter elevation model from the Norwegian Mapping Authority. According to the confusion matrix, 80% of v-valleys were correctly classified, whereas 86% of the u-valleys were identified correctly and 100% of the tested filled up valley samples were correctly classified
- Even though the Gaussian classiffier was not properly trained for the 50 meter Statkart grid, promising results were found, and there is evidence to suggest that the coarser DEM would behave as good as 10 meter Statkart DEM.
- ASTER GDEM offers a great coverage of Earth, with a relatively high resolution. The fundamental problem problem with this elevation model is the high amount of noise. The noise has a negative impact on the Gaussian Classifier, resulting in poor results, even on the training image in Norway. The results are surpassingly better in the Romanian Carpathians, where the majority of valleys were correctly classified as v-valleys.
- The method works well on the SRTM data up untill the kNN classification is applied. Allthough valley segments can be obtained, SRTM DEM has particularly high amount of voids on the valley sides. Even though it is not as noisy as the ASTER GDEM it is assumed that the voids would impact negatively the final Gaussian classification.

The methodology could be improved in the following areas:

- Canyons consist the only type of valley form not identified in this study. Canyons sufficiently large to be identified by a 201x201 structuring element are almost absent in the Norwegian landscape, and haence the classifier cannot be trained, resulting in the classifier being unable to identify at this scale.
- The skeleton splitting algorithm is rudimentary, and may achieve over-segmentation in given areas. To solve this issue, a skeleton pruning algorithm similar to the one developed by Bai and Latecki (2007) could be applied . This could remove only the insignificant skeletons.
- Finding a good way to alternate between flow accumulation and image skeletons would be a considerable advantage for the separation of objects which could be joined by the Black– Top – Hat transform.
- Finding a proper filtering method for the ASTER GDEM in order to ensure that the algorithm is executed without significant information loss regarding the concavity and convexity of the mean curvature could lead to improved overall results in the Multivariate Gaussian classification.
- The methodology requires further development in handling the voids on a DEM, since voids will be identified as valleys and wrongly classified
- Integrating a multi-scale step would be also a valuable feature, in order to find and classify ravines or chutes.

This method could additionally be utilized for change detection, given the two elevation models are correctly co-registered and have the same resolution. The valleys could be separated from other landforms, and valley evolution could be monitored.

The main purpose of this methodology is to identify valleys in Scandinavia. The method can be used on both coarse and high resolution DEM and is independent of location, while the parameters are dependent only on the topography of a country. It is consequently possible to use the methodology on a global scale, provided that enough effort is put in properly training the classifier. Obtaining results on a global scale would help us understand more about landscape development, former ice-sheet configuration and temperature regimes.

7. References

BAATZ, M. & SCHÄPE, A. 2000. Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation. *Angewandte Geographische Informationsverarbeitung XII***,** 12-23.

BAI, X. & LATECKI, L. J. Discrete skeleton evolution. Energy minimization methods in computer vision and pattern recognition, 2007. Springer, 362-374.

BURROUGH, P. A., VAN GAANS, P. F. M. & MACMILLAN, R. A. 2000. High-resolution landform classification using fuzzy k-means. *Fuzzy Sets and Systems,* 113**,** 37-52.

DE MIRA, J. & MAYER, J. Image feature extraction for application of biometric identification of iris-a morphological approach. Computer Graphics and Image Processing, 2003. SIBGRAPI 2003. XVI Brazilian Symposium on, 2003. IEEE, 391-398.

DIKAU, R. 1989. The application of a digital relief model to landform analysis in geomorphology. *Three dimensional applications in geographical information systems***,** 51-77.

DOUGHERTY, G. 2012. *Pattern recognition and classification: an introduction*, Springer Science & Business Media.

DRĂGUŢ, L. & BLASCHKE, T. 2006. Automated classification of landform elements using object-based image analysis. *Geomorphology,* 81**,** 330-344.

DUDA, R. O., HART, P. E. & STORK, D. G. 2001. Pattern classification. 2nd. *Edition. New York.*

ETZELMULLER, B. 2014. *RE: Personal Communication.*

ETZELMULLER, B. 2015. *RE: Personal Communication(e-mail).*

EVANS, I. S. 1972. General geomorphometry, derivatives of altitude, and descriptive statistics. DTIC Document.

FORSYTH, D. 2003. *Computer vision : a modern approach*, Upper Saddle River, N.J, Prentice Hall.

GEOFF, D., AUTHOR, WILLIAM, D., REVIEWER & PH.D 2010. Digital Image Processing for Medical Applications. *Medical Physics,* 37**,** 948.

GERÇEK, D., TOPRAK, V. & STROBL, J. 2011. Object-based classification of landforms based on their local geometry and geomorphometric context. *International Journal of Geographical Information Science,* 25**,** 1011-1023.

GONZALEZ, R. C. & WOODS, R. E. 2002. Digital image processing second edition. *Beijing: Publishing House of Electronics Industry*.

GOUDIE, A. 2004. *Encyclopedia of geomorphology*, Psychology Press.

GUO, G., WANG, H., BELL, D., BI, Y. & GREER, K. 2004. An kNN model-based approach and its application in text categorization. *Computational Linguistics and Intelligent Text Processing.* Springer.

HAAPANEN, R., EK, A. R., BAUER, M. E. & FINLEY, A. O. 2004. Delineation of forest/nonforest land use classes using nearest neighbor methods. *Remote Sensing of Environment,* 89**,** 265-271.

HARALOCK, R. M. & SHAPIRO, L. G. 1991. *Computer and robot vision*, Addison-Wesley Longman Publishing Co., Inc.

HEILBRONNER, R. 2000. Automatic grain boundary detection and grain size analysis using polarization micrographs or orientation images. *Journal of Structural Geology,* 22**,** 969-981.

HENGL, T. & REUTER, H. I. 2009. *Geomorphometry: concepts, software, applications*, Elsevier.

HUGGETT, R. J. 2003. *Fundamentals of geomorphology,* London, Routledge.

KARTVERKET. 2015a. *http://data.kartverket.no/download/content/digital-terrengmodell-10-m-utm-33* [Online]. The norwegian mapping authority. [Accessed 07.29 2015].

KARTVERKET. 2015b. *http://data.kartverket.no/download/content/digital-terrengmodell-50-m-utm-33* [Online]. The Norwegian Mapping Authority. [Accessed 07.29 2015].

KARTVERKET. 2015c. *http://kartverket.no/en/About-The-Norwegian-Mapping-Authority/The-Norwegian-Mapping-Authority/* [Online]. The Norwegian Mapping Authority. 29.07.2015].

KRCHO, J. 1983. Teoretická concepcia a interdisciplinarne aplikacie komplexného digitalneho modelu reliéfu pri modelovanï dvojdimenzionalnych poli. *Geograficky Cas,* 35**,** 265-291.

KRCHO, J. & HAVERLIK, I. 1973. *Morphometric Analyses of Relief on the Basis of Geometrie Aspect of Field Theory-Mathematical Generalisation of Forming Isoline Thematic Maps*

by Computer Exemplified by Morphometrie Analysis of Relief and Dynamics of Relief Insolation, Slovak pedagogical publishers.

LOPES, D. 2012. *Important Concepts in Signal Processing, Image Processing and Data Compression*, University Publications.

LUO, W., PINGEL, T., HEO, J., HOWARD, A. & JUNG, J. 2015. A progressive black top hat transformation algorithm for estimating valley volumes on Mars. *Computers & Geosciences,* 75**,** 17-23.

MACMILLAN, R. A., PETTAPIECE, W. W., NOLAN, S. C. & GODDARD, T. W. 2000. A generic procedure for automatically segmenting landforms into landform elements using DEMs, heuristic rules and fuzzy logic. *Fuzzy Sets and Systems,* 113**,** 81-109.

MATHERON, G. 1967. Eléments pour une théorie des milieux poreux.

MCANDREW, A. An introduction to digital image processing with matlab notes for scm2511 image processing.

MEYER, F. 1979. Iterative image transformations for an automatic screening of cervical smears. *Journal of Histochemistry & Cytochemistry,* 27**,** 128-135.

MITASOVA, H., HOFIERKA, J., ZLOCHA, M. & IVERSON, L. R. 1996. Modelling topographic potential for erosion and deposition using GIS. *International Journal of Geographical Information Systems,* 10**,** 629-641.

NASA. 2015. National Aeronautics and Space Administration: http://www2.jpl.nasa.gov/srtm/. [Accessed 29.07 2015].

NORDGULEN, Ø., BARGEL, T. H., LONGVA, O., OLESEN, O. & OTTESEN, D. 2006. A preliminary study of Lofoten as a potential World Heritage Site based on natural criteria Geological Survey of Norway.

PYTHON, D. T. 2015. https://www.python.org/doc/essays/blurb/. 7.29.2015].

QGIS_DEVELOPMENT_TEAM. 2015. *QGIS Geographic Information System* [Online]. http://www.qgis.org/en/site/about/index.html. [Accessed 7.29.2015.

RICHARDS, J. A. & SPRINGERLINK 2013. Remote Sensing Digital Image Analysis : An Introduction. 5th ed. 2013. ed. Berlin, Heidelberg: Springer Berlin Heidelberg Imprint: Springer.

RODRIGUEZ, F., MAIRE, E., COURJAULT-RADÉ, P. & DARROZES, J. 2002. The Black Top Hat function applied to a DEM: A tool to estimate recent incision in a mountainous

watershed (Estibère Watershed, Central Pyrenees). *Geophysical Research Letters,* 29**,** 9-1-9-4.

ROMSTAD, B. & ETZELMÜLLER, B. 2012. Mean-curvature watersheds: A simple method for segmentation of a digital elevation model into terrain units. *Geomorphology,* 139–140**,** 293-302.

ROSENFELD, A. & PFALTZ, J. L. 1966. Sequential Operations in Digital Picture Processing. *J. ACM,* 13**,** 471-494.

RUSS, J. C. 1998. *The Image Processing Handbook, Third Edition*, Taylor & Francis.

SCHMIDT, J. & HEWITT, A. 2004. Fuzzy land element classification from DTMs based on geometry and terrain position. *Geoderma,* 121**,** 243-256.

SERRA, J. 1986. Introduction to mathematical morphology. *Computer Vision, Graphics, and Image Processing,* 35**,** 283-305.

SHARY, P. 1991. The second derivative topographic method. *The Geometry of the Earth Surface Structures***,** 30-60.

SOILLE, P. 2000. Morphological image analysis applied to crop field mapping. *Image and Vision Computing,* 18**,** 1025-1032.

SOILLE, P. 2013. *Morphological image analysis: principles and applications*, Springer Science & Business Media.

SOLOMON, C. & BRECKON, T. 2011. *Fundamentals of Digital Image Processing: A practical approach with examples in Matlab*, John Wiley & Sons.

TACHIKAWA, T., HATO, M., KAKU, M. & IWASAKI, A. Characteristics of ASTER GDEM version 2. Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International, 24-29 July 2011 2011. 3657-3660.

VAN ZYL, J. J. 2001. The Shuttle Radar Topography Mission (SRTM): a breakthrough in remote sensing of topography. *Acta Astronautica,* 48**,** 559-565.

VERNON, D. 1991. Machine vision-Automated visual inspection and robot vision.

XIE, X., SU, F. & CAI, A. 2005. Ridge-based fingerprint recognition. *Advances in Biometrics.* Springer.

YOUNG, M. & EVANS, I. 1978. Statistical characterization of altitude matrices. Report.

ZEVENBERGEN, L. W. & THORNE, C. R. 1987. Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms,* 12**,** 47-56.

ZHANG, T. & SUEN, C. Y. 1984. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM,* 27**,** 236-239.

Appendix I. Gaussian Training sample for V- valleys. Statkart 10 meter grid.

DEM (UTM z33)	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
6600_4_10m	111	0.0697	0.3804	0.4603	0.0894	0.5128	0.0772	0.4100
	93	0.0313	0.6551	0.2585	0.0549	0.5200	0.0416	0.4384
	112	0.0042	0.7658	0.2203	0.0096	0.4677	0.0508	0.4815
	78	0.0384	0.7428	0.1992	0.0197	0.5112	0.0613	0.4274
	29	0.0619	0.2698	0.4636	0.2014	0.5084	0.0817	0.4099
6700_3_10m	70	0.0098	0.3906	0.4845	0.1150	0.5080	0.0330	0.4591
	23	0.0064	0.3402	0.4367	0.2063	0.5297	0.0368	0.4335
	5	0.0668	0.7694	0.1635	0.0003	0.5116	0.0590	0.4294
6700_2_10m	15	0.0484	0.3647	0.3523	0.2114	0.5151	0.0533	0.4316
	55	0.1363	0.5730	0.1720	0.0993	0.4835	0.1405	0.3760
	49	0.0486	0.4127	0.3450	0.1782	0.4776	0.0654	0.4570
	29	0.1221	0.4091	0.3575	0.0940	0.4407	0.1584	0.4009
6700_4_10m	89	0.0245	0.4686	0.4484	0.0580	0.5598	0.0527	0.3875
	94	0.0425	0.4898	0.3621	0.1038	0.5231	0.0553	0.4216
	46	0.0418	0.5325	0.3705	0.0535	0.5254	0.0467	0.4280
	21	0.0011	0.3496	0.5010	0.1477	0.5267	0.0433	0.4300
6700_1_10m	80	0.0200	0.3467	0.5054	0.1212	0.5082	0.0496	0.4421
	81	0.0038	0.3576	0.4085	0.2047	0.4883	0.0331	0.4786
	23	0.0293	0.7006	0.2592	0.0109	0.5382	0.0686	0.3931
	5	0.0004	0.8184	0.1730	0.0082	0.4482	0.1496	0.4022
6800_3_10m	118	0.0247	0.6695	0.2977	0.0081	0.4957	0.0805	0.4238
	89	0.0641	0.3378	0.4322	0.1646	0.5209	0.0550	0.4241
6800_2_10m	104	0.0035	0.6367	0.3548	0.0050	0.5190	0.0634	0.4176
	103	0.0173	0.6236	0.3510	0.0081	0.5319	0.0634	0.4047
	97	0.0569	0.8745	0.0676	0.0010	0.4888	0.0920	0.4192
6701_4_10m	23	0.0106	0.3650	0.5265	0.0962	0.4908	0.0464	0.4628
	36	0.0088	0.5067	0.3655	0.1116	0.4894	0.0557	0.4550
6802_1_10m	23	0.0148	0.9175	0.0678	0.0000	0.4370	0.0930	0.4700
6902_1_10m	6	0.1097	0.8466	0.0437	0.0000	0.4624	0.1277	0.4099
	18	0.0545	0.8665	0.0788	0.0003	0.4261	0.0812	0.4927
	7	0.0518	0.8035	0.1438	0.0010	0.4713	0.0720	0.4566
	13	0.0227	0.5291	0.4201	0.0281	0.4458	0.0517	0.5025
7003_1_10m	8	0.1212	0.6061	0.2599	0.0128	0.5054	0.0662	0.4284
7708_4_10m	11	0.0783	0.8514	0.0688	0.0015	0.5028	0.0718	0.4254
	24	0.1744	0.6746	0.1476	0.0033	0.3757	0.1677	0.4567
7809_3_10m	5	0.0943	0.8825	0.0231	0.0001	0.4298	0.1953	0.3748

Appendix II. Gaussian Training sample for filled up valleys. Statkart 10 meter grid.

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
6600_4_10m	72	0.3137	0.3838	0.2649	0.0371	0.3837	0.3189	0.2974
	88	0.1704	0.4174	0.3320	0.0794	0.4545	0.1864	0.3591
	91	0.4465	0.2160	0.2995	0.0376	0.3241	0.4419	0.2340
6700_3_10m	67	0.1361	0.3326	0.4513	0.0786	0.4793	0.1456	0.3751
6700_2_10m	24	0.2181	0.7017	0.0729	0.0071	0.4856	0.2292	0.2852
6600_1_10m	30	0.2990	0.5663	0.1329	0.0018	0.4188	0.2974	0.2838
	19	0.4050	0.4335	0.1587	0.0028	0.3426	0.4280	0.2294
6700_4_10m	57	0.0967	0.5160	0.3379	0.0491	0.5157	0.0621	0.4222
	28	0.2134	0.3188	0.3906	0.0762	0.4655	0.1979	0.3367
	76	0.3023	0.5808	0.1073	0.0090	0.4101	0.2829	0.3070
	72	0.2255	0.5320	0.2324	0.0101	0.4234	0.2317	0.3450
	95	0.3158	0.5813	0.0903	0.0123	0.4074	0.2837	0.3090
6700_1_10m	78	0.1482	0.4257	0.3521	0.0724	0.4668	0.1809	0.3523
	98	0.2156	0.4855	0.2382	0.0599	0.4430	0.2297	0.3273
	68	0.1735	0.2474	0.4171	0.1522	0.4565	0.1521	0.3914
6800_3_10m	92	0.1999	0.4306	0.3245	0.0450	0.4843	0.1951	0.3205
	48	0.4043	0.5105	0.0771	0.0078	0.3640	0.3933	0.2427
	29	0.3815	0.4323	0.1584	0.0217	0.3782	0.3848	0.2371
6800_2_10m	38	0.5355	0.2538	0.2004	0.0100	0.2708	0.5399	0.1892
	54	0.4581	0.2486	0.2582	0.0349	0.3247	0.4500	0.2252
	26	0.4582	0.2347	0.2720	0.0351	0.3626	0.4060	0.2314
	13	0.2363	0.3515	0.3311	0.0801	0.4550	0.2628	0.2822
6701_4_10m	13	0.0882	0.4254	0.4360	0.0497	0.5256	0.0804	0.3940
	69	0.2479	0.6335	0.1052	0.0134	0.4873	0.2263	0.2864
6701_1_10m	11	0.4134	0.5331	0.0535	0.0000	0.3396	0.4776	0.1828
	13	0.4589	0.5125	0.0283	0.0004	0.3306	0.4476	0.2218
6801_2_10m	18	0.2643	0.6998	0.0357	0.0002	0.3325	0.4783	0.1892
	32	0.4355	0.2793	0.2365	0.0482	0.3699	0.4548	0.1753
	34	0.1979	0.7145	0.0738	0.0122	0.4457	0.2744	0.2799
6902_1_10m	4	0.1999	0.4914	0.2944	0.0140	0.5000	0.1135	0.3865
7708_4_10m	15	0.3940	0.4928	0.1045	0.0084	0.4207	0.2756	0.3037
7809_3_10m	8	0.3405	0.6588	0.0007	0.0000	0.3847	0.3461	0.2692

Appendix III. Gaussian Training sample for U-valleys. Statkart 10 meter grid.

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
6600_4_10m	69	0.0435	0.7115	0.2047	0.0391	0.5105	0.0715	0.4179
	83	0.0369	0.6213	0.3076	0.0342	0.5139	0.0615	0.4246
	37	0.0758	0.6730	0.2095	0.0416	0.5133	0.0904	0.3963
	30	0.0495	0.5857	0.2976	0.0649	0.5465	0.0681	0.3854
6700_3_10m	58	0.0070	0.4672	0.4234	0.1023	0.5435	0.0437	0.4129
	57	0.0515	0.7304	0.2005	0.0176	0.5257	0.0764	0.3979
	51	0.0216	0.5927	0.3237	0.0594	0.4791	0.0577	0.4632
	18	0.0176	0.7459	0.2252	0.0113	0.5238	0.0545	0.4217
6700_2_10m	51	0.0669	0.6885	0.2062	0.0384	0.5643	0.1458	0.2899
6600_1_10m	27	0.0639	0.7222	0.1754	0.0385	0.5470	0.1174	0.3356
	39	0.0574	0.6999	0.2156	0.0269	0.5433	0.0747	0.3820
	20	0.0545	0.8107	0.1226	0.0121	0.5562	0.1694	0.2745
	8	0.1426	0.7651	0.0859	0.0063	0.5140	0.2163	0.2697
	44	0.0333	0.5371	0.3444	0.0838	0.5657	0.0545	0.3798
6700_4_10m	80	0.0181	0.6877	0.2758	0.0184	0.5502	0.0827	0.3670
	58	0.0448	0.7107	0.2332	0.0113	0.5087	0.0876	0.4037
	26	0.0288	0.4618	0.3842	0.1191	0.5619	0.0852	0.3529
	34	0.1153	0.5935	0.2421	0.0491	0.5291	0.1191	0.3518
	39	0.0590	0.6335	0.2635	0.0430	0.5489	0.0675	0.3836
	86	0.0657	0.7393	0.1765	0.0183	0.5139	0.0949	0.3912
	87	0.1301	0.8027	0.0658	0.0014	0.4929	0.1082	0.3988
6700_1_10m	86	0.0091	0.7021	0.2788	0.0101	0.5602	0.0878	0.3520
	77	0.0391	0.4966	0.3975	0.0635	0.5632	0.0952	0.3416
	72	0.0693	0.4642	0.3909	0.0713	0.5749	0.1067	0.3185
	10	0.0284	0.3576	0.4581	0.1477	0.5355	0.0586	0.4059
6800_3_10m	125	0.0157	0.3767	0.4608	0.1410	0.5257	0.0428	0.4315
	120	0.0473	0.5026	0.3478	0.0982	0.5613	0.0796	0.3590
	122	0.0420	0.4457	0.3738	0.1364	0.5790	0.0702	0.3508
	55	0.1020	0.8034	0.0900	0.0045	0.5061	0.1069	0.3869
	19	0.1329	0.7991	0.0673	0.0008	0.4798	0.1781	0.3421
	9	0.0470	0.6174	0.3052	0.0301	0.5541	0.0973	0.3486
6800_2_10m	52	0.0229	0.4426	0.4152	0.1167	0.5638	0.0659	0.3703
	37	0.0231	0.4184	0.4514	0.1048	0.5433	0.0736	0.3831
	72	0.0049	0.4464	0.5217	0.0270	0.5561	0.0552	0.3886
	70	0.1050	0.4179	0.4146	0.0616	0.5589	0.1144	0.3266
	48	0.0495	0.7484	0.1913	0.0108	0.5109	0.1070	0.3821

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
6701_4_10m	44	0.0071	0.3461	0.4929	0.1508	0.5736	0.0478	0.3786
	46	0.0201	0.3117	0.5115	0.1472	0.5290	0.0512	0.4199
	14	0.0818	0.4549	0.3914	0.0712	0.5216	0.0949	0.3834
	22	0.0319	0.7708	0.1870	0.0103	0.4893	0.1147	0.3960
	66	0.1679	0.7780	0.0487	0.0053	0.5052	0.2321	0.2627
	72	0.0893	0.8158	0.0907	0.0041	0.5135	0.1397	0.3468
6701_1_10m	12	0.2060	0.7922	0.0018	0.0000	0.4821	0.2406	0.2773
6802_1_10m	9	0.1356	0.7839	0.0805	0.0000	0.5017	0.0982	0.4002
7505_4_10m	30	0.0342	0.4544	0.4633	0.0481	0.5600	0.0573	0.3827

Appendix IV.A Gaussian Training sample for V-valleys. ASTER GDEM.

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
377911208	213	0.0542	0.8038	0.1090	0.0292	0.4901	0.0461	0.4638
	253	0.0128	0.9258	0.0614	0.0000	0.4966	0.0500	0.4534
	193	0.0114	0.5882	0.3352	0.0647	0.5033	0.0288	0.4678
	297	0.0209	0.7674	0.2030	0.0087	0.4862	0.0397	0.4741
	286	0.0119	0.4777	0.4180	0.0917	0.5066	0.0238	0.4696
	277	0.0470	0.8580	0.0920	0.0029	0.4848	0.0495	0.4657
	273	0.0245	0.7201	0.2420	0.0134	0.5144	0.0356	0.4500
	503	0.0357	0.8557	0.1041	0.0045	0.4927	0.0425	0.4649
	476	0.0452	0.9059	0.0484	0.0005	0.4883	0.0464	0.4653
	455	0.0171	0.8581	0.1219	0.0030	0.4887	0.0344	0.4770
	217	0.0269	0.7456	0.1947	0.0326	0.5035	0.0388	0.4577

Appendix IV.B Gaussian Training sample for U-valleys. ASTER GDEM

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
377911208	212	0.0149	0.6443	0.2896	0.0506	0.5068	0.0278	0.4654
	237	0.0391	0.8613	0.0932	0.0064	0.4983	0.0376	0.4641
	243	0.0379	0.8542	0.0928	0.0144	0.5055	0.0417	0.4528
	268	0.0664	0.9111	0.0225	0.0000	0.4902	0.0465	0.4633
	323	0.0189	0.6198	0.3002	0.0606	0.4997	0.0293	0.4711
	353	0.0254	0.7854	0.1814	0.0078	0.5015	0.0387	0.4599
	441	0.0833	0.7883	0.1214	0.0070	0.5055	0.0492	0.4453
	426	0.0466	0.8314	0.1104	0.0116	0.4960	0.0446	0.4595
	464	0.0623	0.8538	0.0750	0.0088	0.5037	0.0516	0.4446
	473	0.0256	0.6988	0.2386	0.0367	0.5081	0.0420	0.4499
	546	0.0657	0.8810	0.0503	0.0028	0.4907	0.0471	0.4622
	554	0.0482	0.8098	0.1286	0.0130	0.4967	0.0395	0.4637

Appendix IV.C Gaussian Training sample for filled up valleys. ASTER GDEM

DEM	Object	S < 5	5 < S < 30	30< S < 50	50< S < 70	Negative curvature	No curvature	Positive Curvature
377911208	186	0.0150	0.5939	0.3305	0.0605	0.5021	0.0298	0.4681
	161	0.0790	0.9005	0.0204	0.0001	0.4885	0.0623	0.4492
	309	0.0989	0.8063	0.0883	0.0065	0.5009	0.0525	0.4467
	459	0.3961	0.4954	0.0931	0.0139	0.3338	0.3901	0.2761
	478	0.4788	0.3619	0.1402	0.0186	0.2775	0.4746	0.2479
	324	0.0579	0.6665	0.2382	0.0366	0.4999	0.0369	0.4631
457300359	23	0.5633	0.4170	0.0165	0.0032	0.2886	0.4774	0.2341
	29	0.0971	0.8674	0.0345	0.0010	0.4992	0.0523	0.4484
	65	0.0811	0.8966	0.0223	0.0000	0.4875	0.0596	0.4528
	38	0.0488	0.6254	0.2874	0.0384	0.5000	0.0351	0.4649
	36	0.0469	0.6683	0.2420	0.0424	0.5041	0.0351	0.4609

Appendix V. Samples for confusion matrix Statkart 10 meter grid

DEM	Obj	Classified	Actual Class
7707_3_10	137	V	V
	138	V	V
	114	Filled Up	Filled Up
	146	U	U
	83	Filled Up	Filled Up
	72	U	U
	35	U	U
	36	U	U
	3	Filled Up	Filled Up
	13	V	V
	19	V	V
7505_1_10	118	Filled Up	Filled Up
	104	U	U
	123	V	V
	105	U	U
7405_4_10	95	Filled Up	Filled Up
	112	U	U
	141	Filled Up	Filled Up
	9	V	V
	1	Filled Up	Filled Up
	66	V	V
7002_3_10	37	Filled Up	Filled Up
	18	U	U
	59	U	U
	55	U	U
	87	U	U
	103	Filled Up	Filled Up
7002_3_10	70	U	U
	50	Filled Up	Filled Up
	21	V	V
	25	V	V
	16	Filled Up	Filled Up
	67	V	V
	91	V	V

DEM	Obj	Classified	Actual Class
6902_4_10	7	Filled Up	Filled Up
	31	U	Filled Up
	28	U	U
	39	Filled Up	Filled Up
	23	U	U
	18	Filled Up	Filled Up
	8	Filled Up	V
	6	Filled Up	Filled Up
	59	V	V
6800_1_10	27	U	U
	15	U	U
	48	Filled Up	Filled Up
	53	V	U
	76	U	U
	77	Filled Up	Filled Up
	105	V	V
	122	V	U
	117	V	Filled Up
	154	V	V
	182	V	V
	136	Filled Up	Filled Up
	126	Filled Up	Filled Up
	124	Filled Up	Filled Up
	156	V	V
	225	V	U
	257	Filled Up	Filled Up
6600_2_10	38	Filled Up	Filled Up
	52	Filled Up	Filled Up
	98	U	U
	108	U	U
	126	U	V
	129	Filled Up	Filled Up
	113	U	U
	112	U	U
	117	Filled Up	Filled Up
	101	V	V
	96	V	V
	45	U	Filled Up
	40	Filled Up	Filled Up

DEM	Obj	Classified	Actual Class
65m1_10_10	35	U	U
	39	V	V
	24	U	U
	14	Filled Up	Filled Up
7003_3_10	5	V	V
	16	Filled Up	Filled Up
	20	U	V
	39	V	V
6901_4_10	69	V	V
	85	V	V
	122	Filled Up	Filled Up
	119	U	U
	164	V	Filled Up
	36	U	U
6800_4_10	22	V	V
	100	Filled Up	Filled Up
	116	U	U
	83	V	U

Appendix VI. Sample for confusion matrix ASTER GDEM.

DEM	Object	Classified	Actual Class
377911208	306	U	Filled Up
	313	V	V
	347	Filled Up	Filled Up
	340	V	U
	324	Filled Up	Filled Up
	329	V	V
	361	U	U
	401	Filled Up	Filled Up
	476	U	U
	503	U	V
	553	U	Filled Up
	508	V	U
	515	U	U
	523	V	U
	516	Filled Up	Filled Up
	464	U	U
	472	V	U
	430	V	Filled Up
	399	Filled Up	Filled Up
	459	Filled Up	Filled Up
	345	V	V
	216	V	V

DEM	Object	Classified	Actual Class
377911208	176	U	U
	85	V	U
	124	V	V
	82	V	U
	134	Filled Up	Filled Up
	195	V	Filled Up
	147	V	V
	15	V	U
	46	U	U
	54	U	U
	16	U	U
	21	U	U
	76	U	Filled Up
	59	U	U
	93	U	Filled Up
	80	Filled Up	Filled Up
	86	U	U
	191	Filled Up	Filled Up
	245	Filled Up	Filled Up
	324	Filled Up	Filled Up
	309	Filled Up	Filled Up
	396	Filled Up	V
	511	Filled Up	U

APPENDIX A. Code for, Filtering Top – Hat transform, Skeletons, kNN classification

```
import glob
from PyQt4.QtCore import *
from PyQt4.QtGui import *
import os
from osgeo import gdal
import numpy as np
import osr
import sys
import math
import glob
import os
import time
import numpy as np
from PIL import Image
import sys
import math
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import cv2
from scipy import ndimage
import copy
#import collection
import skimage.morphology
import numexpr as ne
from numexpr import evaluate
from mpl_toolkits.mplot3d import Axes3D
import scipy.ndimage
# define parameters



path = "/uio/hume/student-u10/robertna/Mesteri/*.tif"
files=glob.glob(path)
path2 = "/uio/hume/student-u10/robertna/Mesteri/"
files=glob.glob(path)
sigma=15 # size of primary gaussian filter
sigma2=10#secondary
strelem=(201,201)
elev_th=-100
size_th=20000
stream_th=100
stepval_1=4000
th_val3=20000
conectivity=np.array([[1,1,1],[1,1,1],[1,1,1]])
no_nbrs=1
stepval2=4000
# function that removes small elements from the image as an input it takes a connected component image, the value that needs to be checked, and the value to be replaced to
def remsmall(lab_img,th_val,no_to_replace):
   conectivity=np.array([[1,1,1],[1,1,1],[1,1,1]])
```

```
    un=np.unique(lab_img)
    un = un[~np.isnan(un)]
    un=np.asarray(un,dtype="float")
    numel=len(un)
    for i in range(numel):
        itemindex = np.where(lab_img==un[i])
        r=itemindex[0]
        c=itemindex[1]
        if len(r)<th_val:
            lab_img[r,c]=no_to_replace

    un1=np.unique(lab_img)
    un1 = un1[~np.isnan(un1)]
    un1=np.asarray(un1,dtype="float")
    numel=len(un1)
    fin_label=1
    for i in range(1,numel):
        lab_img[lab_img==un1[i]]=fin_label
        fin_label=fin_label+1

    return lab_img

    return lab_img

def split(img1,th_val):
#splits the skeletons necessary for the knn classification
    test2=np.zeros((img1.shape[0],img1.shape[1]))
    conectivity=np.array([[1,1,1],[1,1,1],[1,1,1]])
    for i in range(2,img1.shape[0]-2):
        for j in range(2,img1.shape[1]-2):
            if img1[i,j]==1 and sum(sum(img1[i-2:i+3,j-2:j+3]))>6:
                test2[i,j]=1
    img1=img1-test2
    con_band1, nb_labels = ndimage.label(img1,structure=conectivity)
    band1_small=remsmall(con_band1,th_val,0)
    band2=copy.deepcopy(band1_small)
    band2[band2>0]=1
    test2=np.zeros((img1.shape[0],img1.shape[1]))
    for i in range(2,img1.shape[0]-2):
        for j in range(2,img1.shape[1]-2):
            if band2[i,j]==1 and sum(sum(band2[i-2:i+3,j-2:j+3]))>6:
                test2[i,j]=1
    band2=band2-test2
    con_band1, nb_labels = ndimage.label(band2,structure=conectivity)
    return con_band1


def knn(img,lab_img,no_nbrs,pix_val_at_point,step):
# knn classification. The function is adapted to find the first nearest neighbor.
    stepval2=step
    t1=time.time()
    print "t1 started"+str(t1)
        classes=np.where(lab_img>0)
    x_c=classes[0]
    y_c=classes[1]
```

```
    #get values
    vals=lab_img[x_c,y_c]
    #find pixel postions
    pixels=np.where(img==pix_val_at_point)
    x_p=pixels[0]
    y_p=pixels[1]
    #prealocate image size
    knn_img=np.zeros((img.shape[0],img.shape[1]))
    maxval=len(x_p)
    if step>maxval:
       step=maxval
    items=range(0,maxval,step)
    items.append(maxval)
    items=np.asarray(items)
    class_x=np.array([x_c,]*step).T
    class_y=np.array([y_c,]*step).T
    valval=np.array([vals]*step).T


    for i in range(len(items)-1):
       if items[i+1]==maxval:
            class_x=np.array([x_c,]*(items[-1]-items[-2])).T
            class_y=np.array([y_c,]*(items[-1]-items[-2])).T
            valval=np.array([vals,]*(items[-1]-items[-2])).T
       pix_x=x_p[items[i]:items[i+1]]
       pix_y=y_p[items[i]:items[i+1]]
       rownums=np.arange(pix_x.shape[0])
       difference=evaluate('sqrt((pix_x-class_x)*(pix_x-class_x)+(pix_y-class_y)*(pix_y-class_y))')

       mins=np.argmin(difference, axis=0)
       xx=valval[mins,rownums[:,None].T]
       xx2=xx[0,:]
       knn_img[pix_x,pix_y]=xx2
       del(difference)
       del(xx)
       del(xx2)

    return knn_img

for file in files:
    # looping through the elevation models to batch process
    fileInfo = QFileInfo(file)
    baseName = fileInfo.baseName()
    ds = gdal.Open(file)
    band = ds.GetRasterBand(1)
    band1 = band.ReadAsArray()
    cols = band1.shape[1]
    rows = band1.shape[0]

    temp=scipy.ndimage.gaussian_filter(band1,sigma)
    print "Primary filtering finished"
    closed=scipy.ndimage.morphology.grey_closing(temp,size=strelem)
    print "Gray level closing finished"
    temp2=scipy.ndimage.filters.gaussian_filter(closed,sigma2)
    print "Secondary filtering finished"
```

```
    temp1=temp-temp2
    temp1[temp1>0]=0
    temp1[temp1>elev_th]=0
    temp1[temp1<elev_th]=1
    con_band2, nb_labels= ndimage.label(temp1,structure=conectivity)
    img1=remsmall(con_band2,size_th,0)
    print "Small areas removed from image"
    img1[img1>0]=1
    skelet=skimage.morphology.skeletonize(img1)
    print type(skelet)
    print "Image has ben skeletonized"
    con_band1=split(skelet,stream_th)
    print "Lines have been splitted"
    classified=knn(img1,con_band1,no_nbrs,1,stepval_1)
    print "Image has been classified"

    geotransform = ds.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]

    var=baseName+'mean_fil'+str(sigma)+'.tif'
    var2=os.path.join(path2,str(var))
    driver = gdal.GetDriverByName('GTiff')
    outRaster = driver.Create(var2, cols, rows, 1, gdal.GDT_Float32)
    outRaster.SetGeoTransform((originX, pixelWidth, 0, originY, 0, pixelHeight))
    outband = outRaster.GetRasterBand(1)
    outband.WriteArray(classified)
    outRasterSRS = osr.SpatialReference()
    outRasterSRS.ImportFromEPSG(32633)
    outRaster.SetProjection(outRasterSRS.ExportToWkt())
    outband.FlushCache()
    print "Another one bytes the dust"



print "Mission accomplished"
```

APPENDIX B. Code for Multivariate Gaussian classification

```
import glob
from PyQt4.QtCore import *
from PyQt4.QtGui import *
from qgis.core import *
import qgis.utils
import os
from osgeo import gdal
import numpy as np
import osr
import sys
import math
sys.path.append('C:/Python27/Lib/site-packages')
import matplotlib.pyplot as plt
import time
import scipy.misc
import cv2
from scipy import ndimage
import copy
import skimage.morphology

from mpl_toolkits.mplot3d import Axes3D
import scipy.ndimage
import statsmodels.api as sm
from numpy.linalg import inv
from scipy import linalg
from numpy import linalg
from PIL import Image

path7="D:/Master/classiffied2/"
path = "D:/Master/with_small/*.tif"
files=glob.glob(path)
path2="D:/Master/slope/*.tif"
slope=glob.glob(path2)
path6="D:/Master/curv/*.tif"
curvat=glob.glob(path6)
path8="D:/Master/DEM/*.tif"
dem=glob.glob(path8)
V_valleys=np.loadtxt("D:/V_Valleys2.txt")
U_valleys=np.loadtxt("D:/U_Valleys2.txt")
F_U_valleys=np.loadtxt("D:/F_U_Valleys2.txt")


def samples(dem,values,files,slope,curvat):
# function that takes the valley objects/segments and retrieves the values for the features
   valleys=files[dem]
   slopes=slope[dem]
   curvature=curvat[dem]
   slope_th=np.array([3,30,50,70])
   curv_th=np.array([-0.0005,0.0005])
```

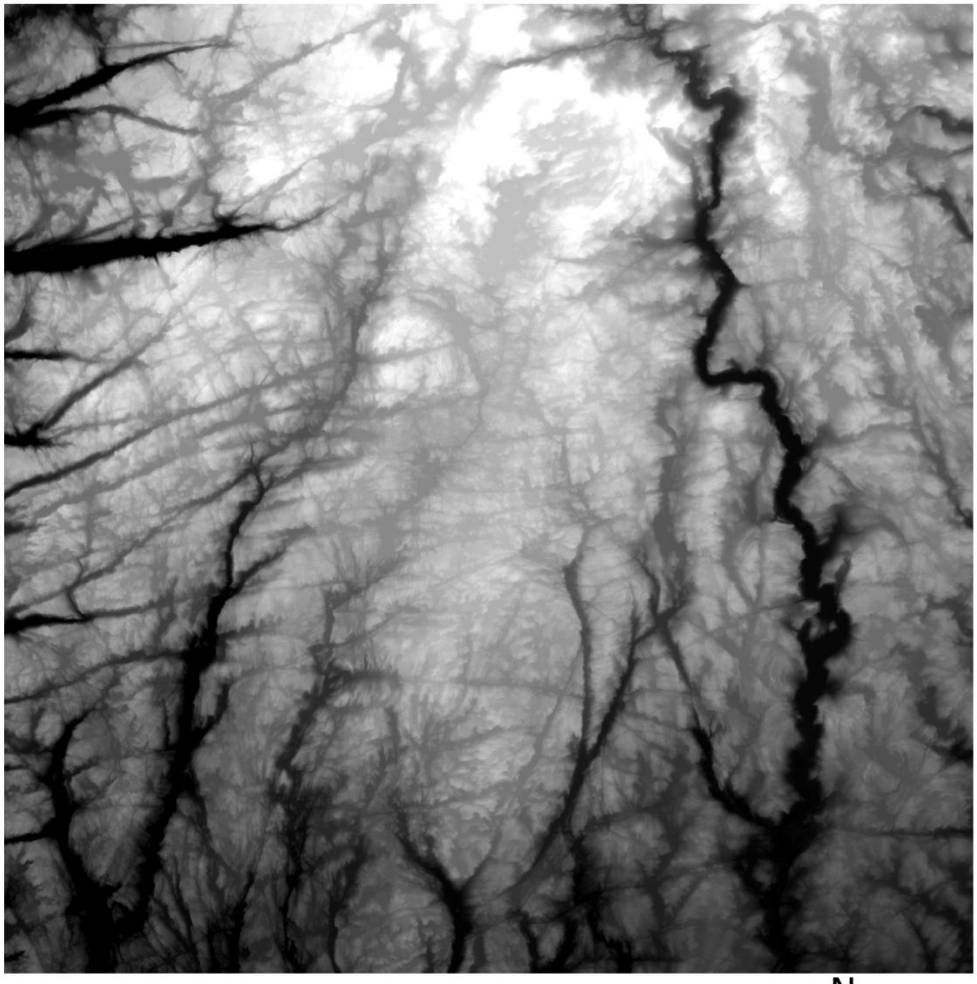
```
valleys= Image.open(valleys)
valleys = np.array(valleys)
slopes= Image.open(slopes)
slopes = np.array(slopes)
slopes=np.degrees(slopes)
curvature= Image.open(curvature)
curvature = np.array(curvature)
tab=np.zeros((values.shape[0],9))
j=0
for i in values:
    dummy=copy.deepcopy(valleys)
    dummy[dummy!=i]=0
    dummy[dummy==i]=1
    ind=np.where(dummy==1)
    row=ind[0]
    col=ind[1]
    #Slope
    slop=slopes[row,col]
    total=len(slop)
    #th1 slope
    sl_th1=slop[slop<slope_th[0]]
    sl_th1=float(len(sl_th1))/float(total)
    #th2 slope
    sl_th2=slop[slop>slope_th[0]]
    sl_th2=sl_th2[sl_th2<slope_th[1]]
    sl_th2=float(len(sl_th2))/float(total)
    #th3 slope
    sl_th3=slop[slop>slope_th[1]]
    sl_th3=sl_th3[sl_th3<slope_th[2]]
    sl_th3=float(len(sl_th3))/float(total)
    # th4 slope
    sl_th4=slop[slop>slope_th[2]]
    sl_th4=sl_th4[sl_th4<slope_th[3]]
    sl_th4=float(len(sl_th4))/float(total)
    # th5 slope
    tab[j,0]=i
    tab[j,1]=sl_th1
    tab[j,2]=sl_th2
    tab[j,3]=sl_th3
    tab[j,4]=sl_th4
    # curvature
    curv=curvature[row,col]
    #th1 curv negative
    curv_th1=curv[curv<curv_th[0]]
    curv_th1=float(len(curv_th1))/float(total)
    #th2 no curv
    curv_th2=curv[curv>curv_th[0]]
    curv_th2=curv_th2[curv_th2<curv_th[1]]
    curv_th2=float(len(curv_th2))/float(total)
    # th3 pozitive curv
    curv_th3=curv[curv>curv_th[1]]
    curv_th3=float(len(curv_th3))/float(total)
    tab[j,5]=curv_th1
    tab[j,6]=curv_th2
    tab[j,7]=curv_th3
```

```
        j=j+1
        print i
        del sl_th1,sl_th2,sl_th3,sl_th4,curv_th1,curv_th2,curv_th3,curv,slop
    del(dummy)
    tab=np.round(tab,decimals=4)
    np.set_printoptions(suppress=True)
    return(tab)

def means_covs(V_valleys,U_valleys,F_U_valleys):
# function which takes the samples and calculates the data necessary for the training of the Gaussian classifier
    means=np.zeros((3,V_valleys.shape[1]-1))
    covariance=np.empty((V_valleys.shape[1]-1,V_valleys.shape[1]-1,3))
    means[0,:]=np.mean(V_valleys[:,:-1],axis=0)
    means[1,:]=np.mean(U_valleys[:,:-1],axis=0)
    means[2,:]=np.mean(F_U_valleys[:,:-1],axis=0)
    covariance[:,:,0]=np.cov(V_valleys[:,:-1],rowvar=0)
    covariance[:,:,1]=np.cov(U_valleys[:,:-1],rowvar=0)
    covariance[:,:,2]=np.cov(F_U_valleys[:,:-1],rowvar=0)
    return means,covariance


mean,cov=means_covs(V_valleys,U_valleys,F_U_valleys)
for i in range(len(files)):
# looping through images in order to batch process
        fileInfo = QFileInfo(dem[i])
    baseNa= fileInfo.baseName()
    rlayer = QgsRasterLayer(dem[i], baseNa)
    ds2= gdal.Open(dem[i])
    DEM= ds2.GetRasterBand(1)
    DEM= DEM.ReadAsArray()


    fileInfo = QFileInfo(files[i])
    baseName = fileInfo.baseName()
    rlayer = QgsRasterLayer(files[i], baseName)
    ds = gdal.Open(files[i])
    band = ds.GetRasterBand(1)
    band1 = band.ReadAsArray()

    cls=np.unique(band1)
    data=samples(i,cls[1:],files,slope,curvat)
    data=data[:,:-1]
    d=data.shape[1]/2
    x=cov.shape[2]
    cla=np.zeros((band1.shape[0],band1.shape[1]))
    for ii in range(data.shape[0]-1):
        prob=np.empty(x)
        for j in range(x):
            prior=1/(((2*math.pi)**d)*(np.linalg.det(np.matrix(cov[:,:,j]))**0.5))
            va=(np.matrix((data[ii,1:]-mean[j,:]))).dot((np.matrix(cov[:,:,j])).I)
            va2=va.dot(((data[ii,1:]-mean[j,:]).T))
            prob[j]=prior*math.exp(-0.5*va2)
        val=prob.argmax()+1
        indexes=np.where(band1==data[ii,0])
        r=indexes[0]
```

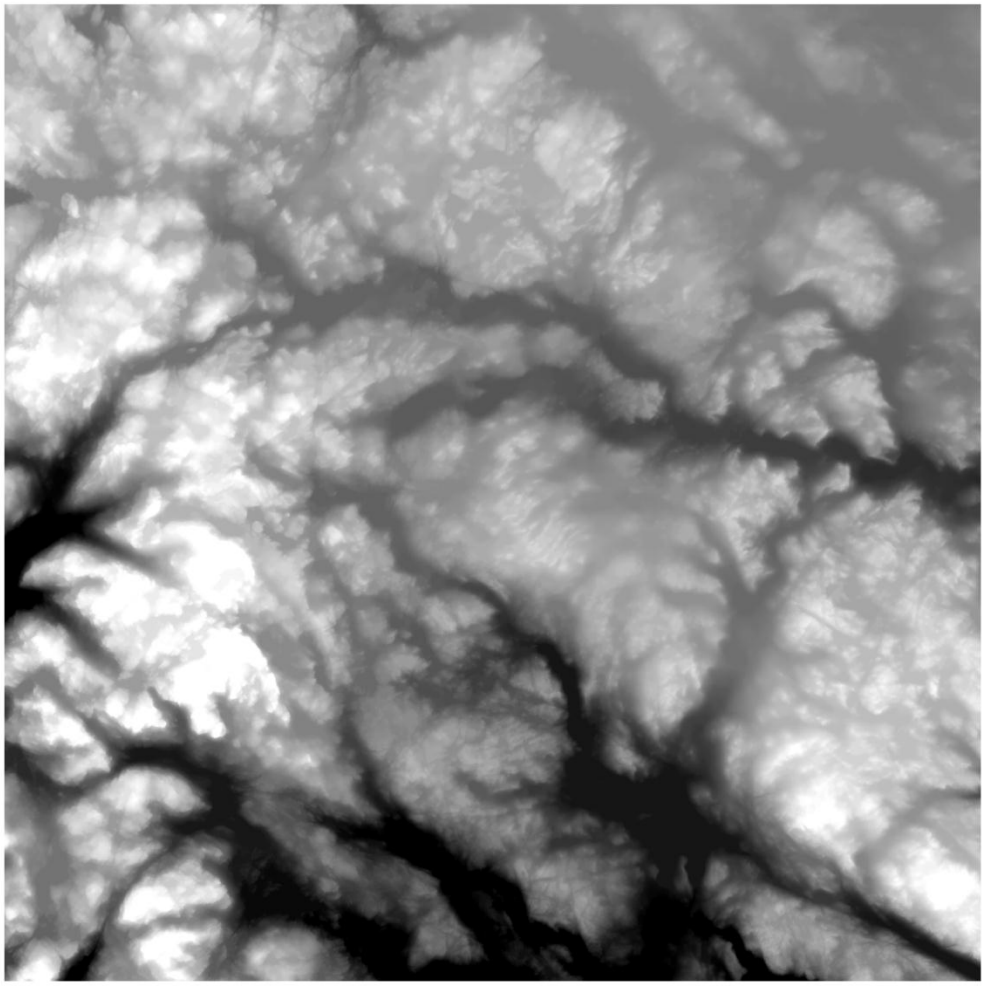
```
        c=indexes[1]
        dem_val=DEM[r,c]
        instances=list(dem_val).count(0)
        instances=instances/float(len(dem_val))
        if instances>0.05:
            cla[r,c]=5
        else:
            cla[r,c]=val
    var=baseName+'class.tif'
    var2=os.path.join(path7,var)
    cols = band1.shape[1]
    rows = band1.shape[0]
    geotransform = ds.GetGeoTransform()
    originX = geotransform[0]
    originY = geotransform[3]
    pixelWidth = geotransform[1]
    pixelHeight = geotransform[5]
    driver = gdal.GetDriverByName('GTiff')
    outRaster = driver.Create(var2, cols, rows, 1, gdal.GDT_Float32)
    outRaster.SetGeoTransform((originX, pixelWidth, 0, originY, 0, pixelHeight))
    outband = outRaster.GetRasterBand(1)
    outband.WriteArray(cla)
    outRasterSRS = osr.SpatialReference()
    outRasterSRS.ImportFromEPSG(32633)
    outRaster.SetProjection(outRasterSRS.ExportToWkt())
    outband.FlushCache()
    print "image"+str(i)+" out of " + str(len(files))
```

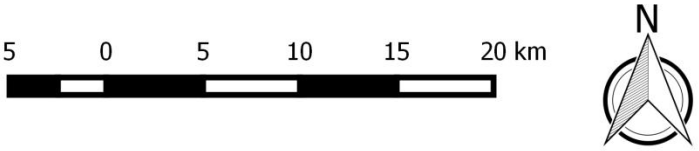
Appendix C. DEMs used in the study

DEM 6500_50m_33

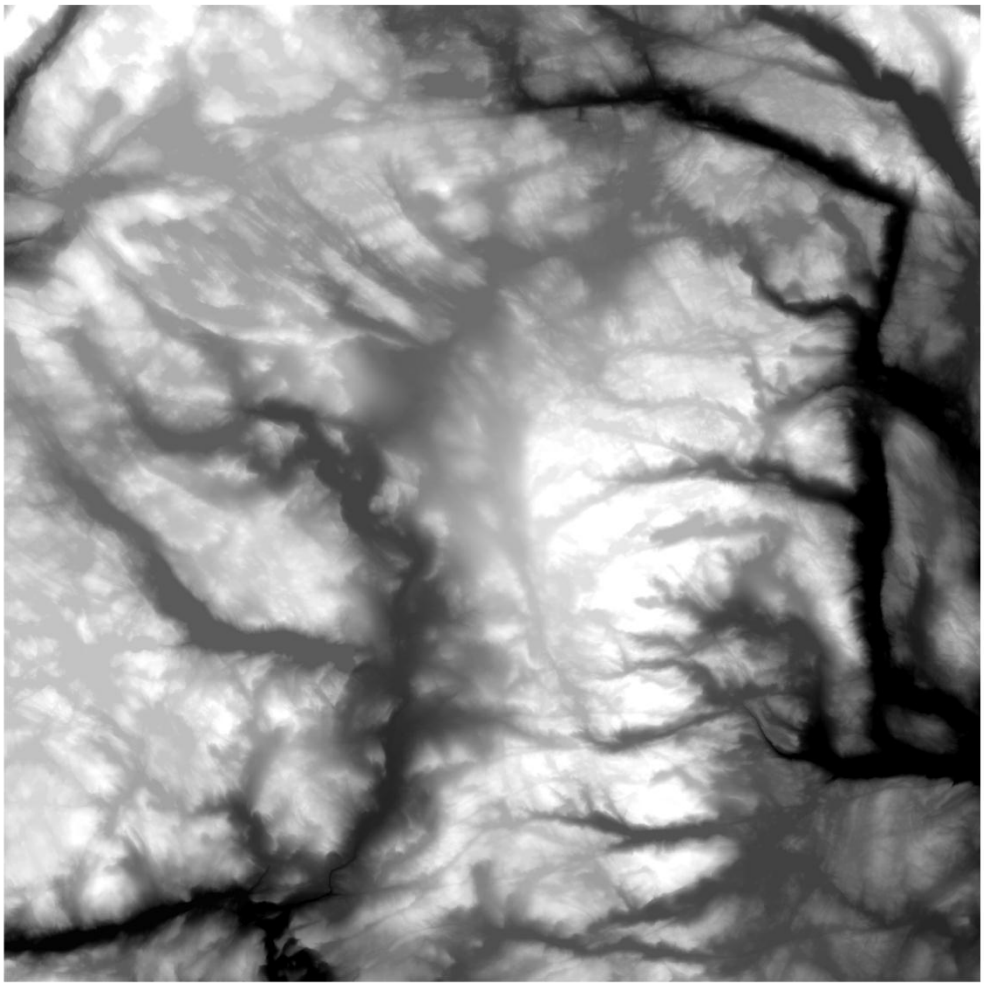


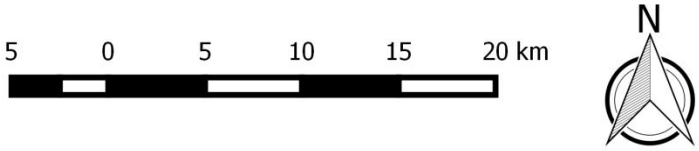
DEM 6600_1_10



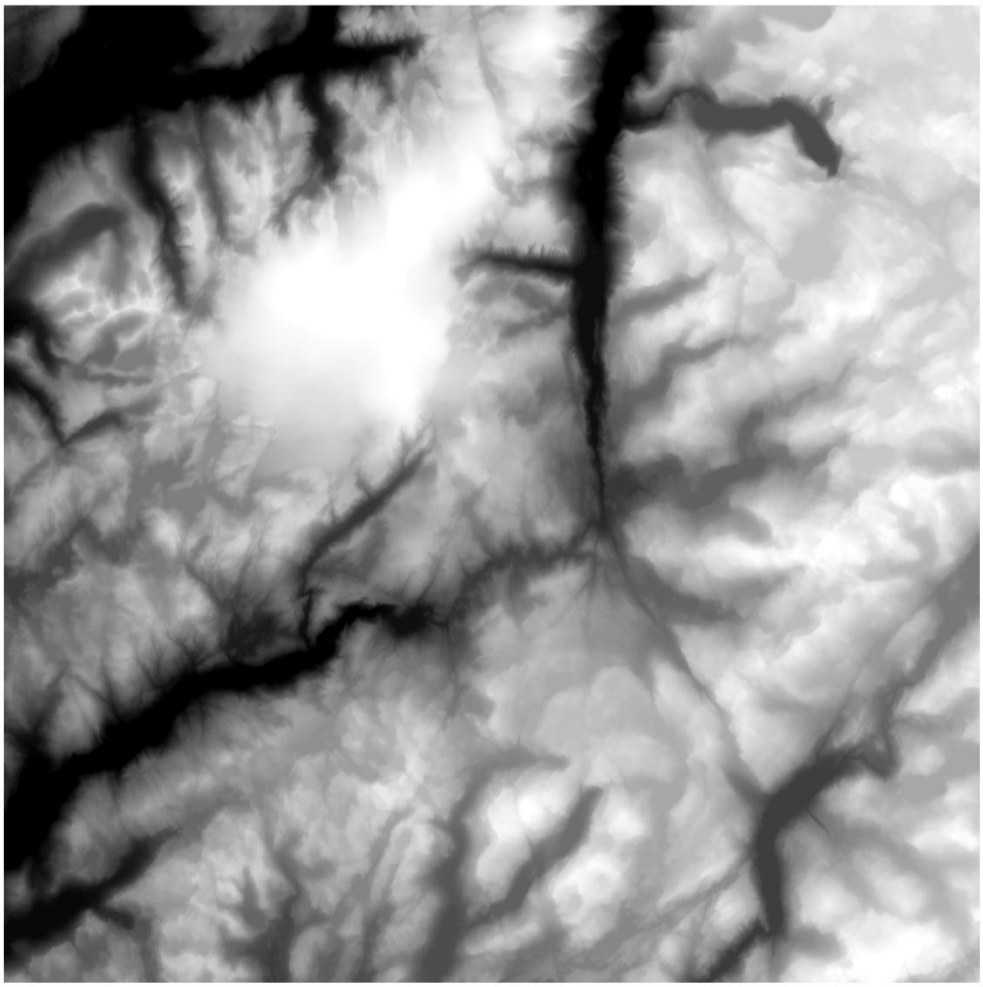


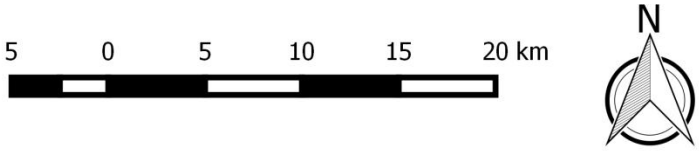
DEM 6600_2_10



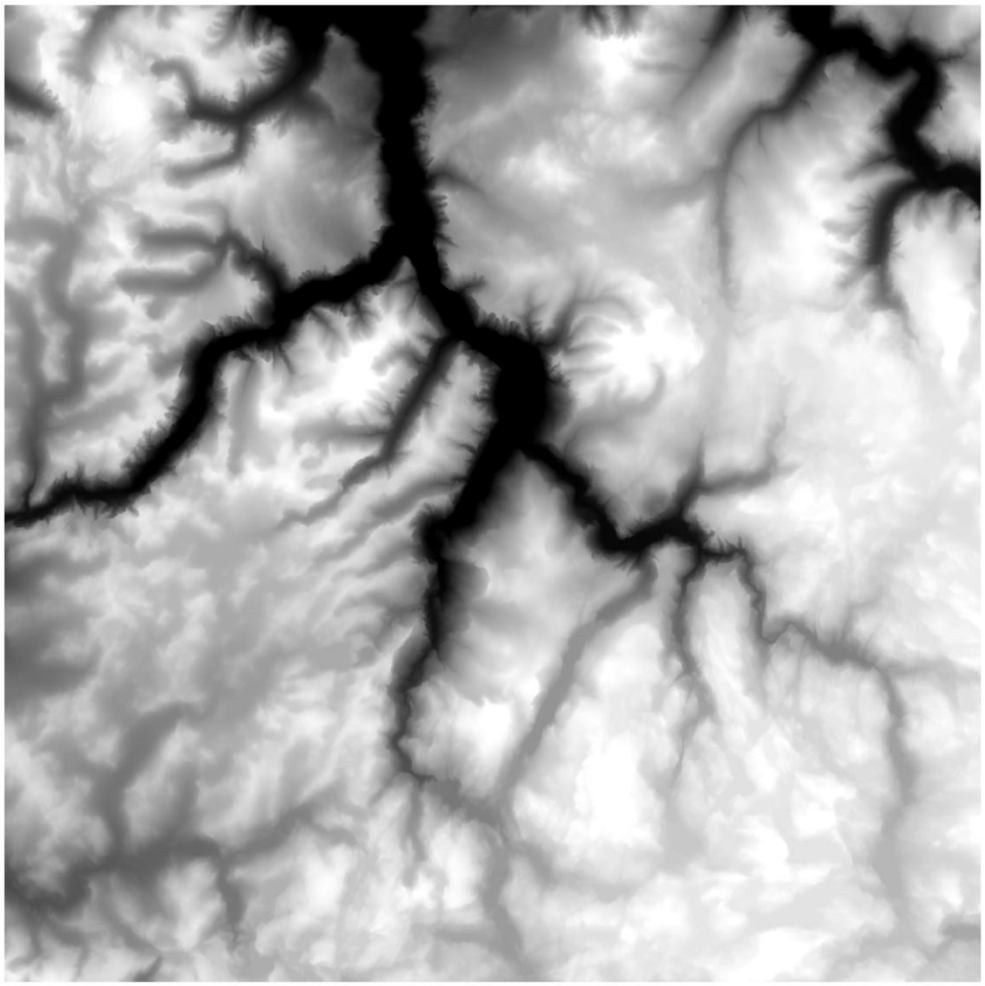


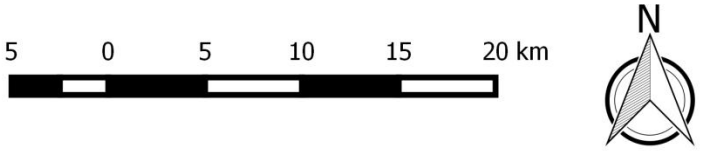
DEM 6600_4_10



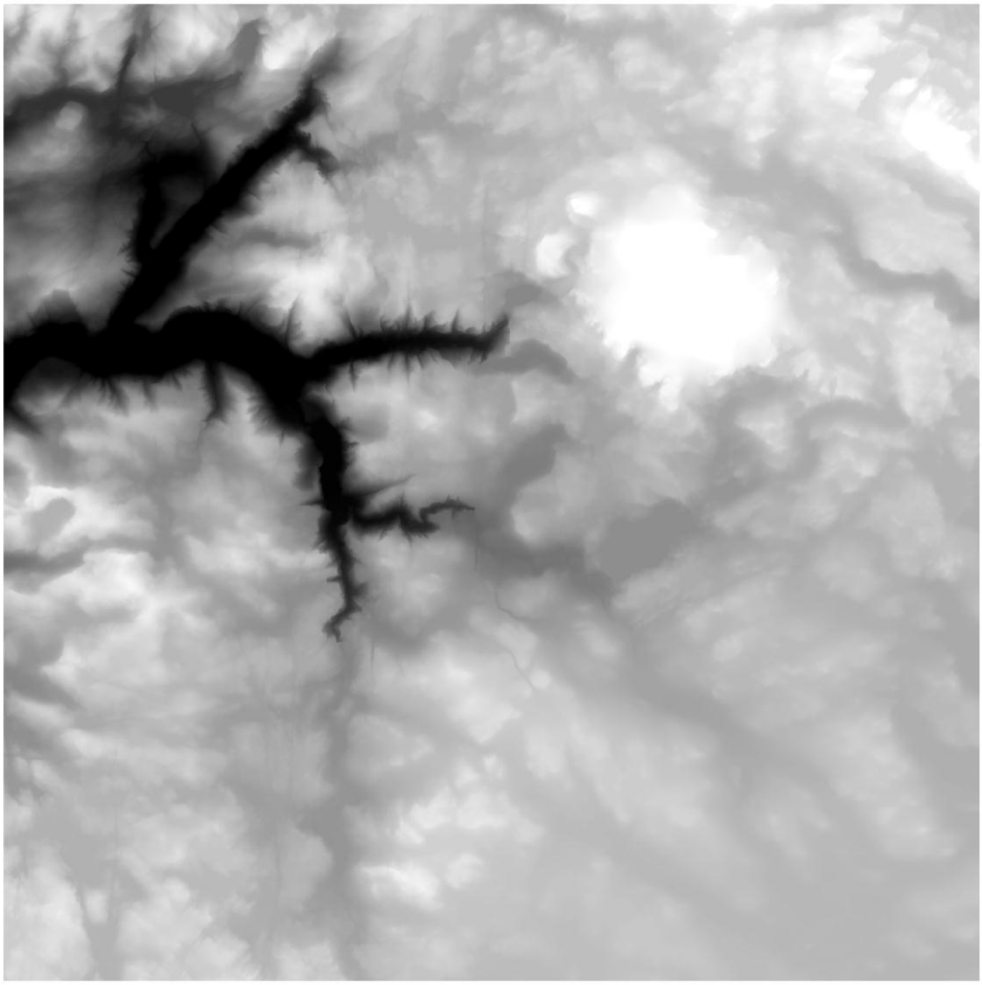


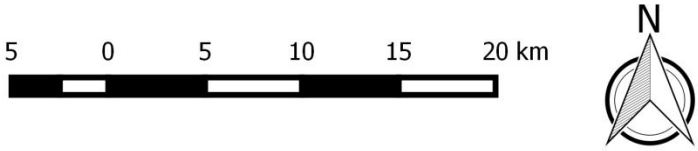
DEM 6700_1_10



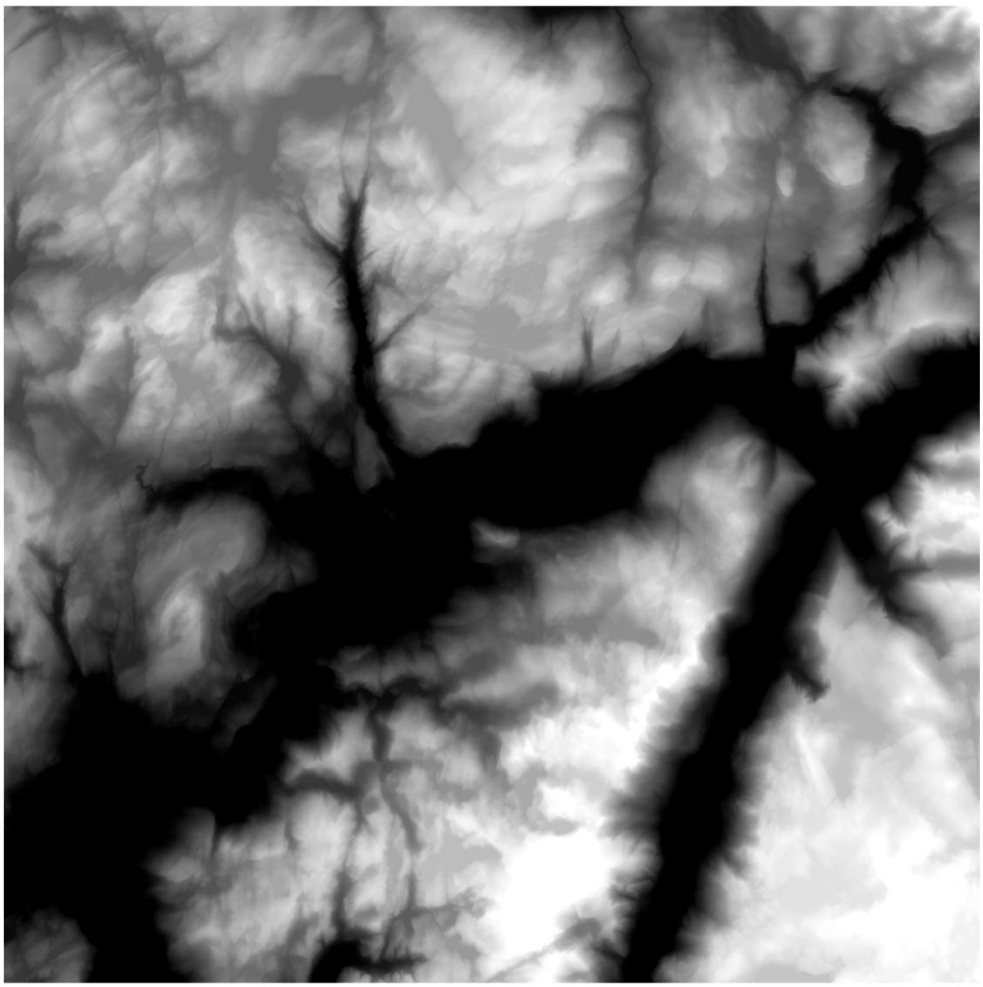


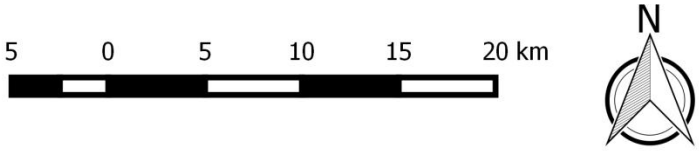
DEM 6700_2_10



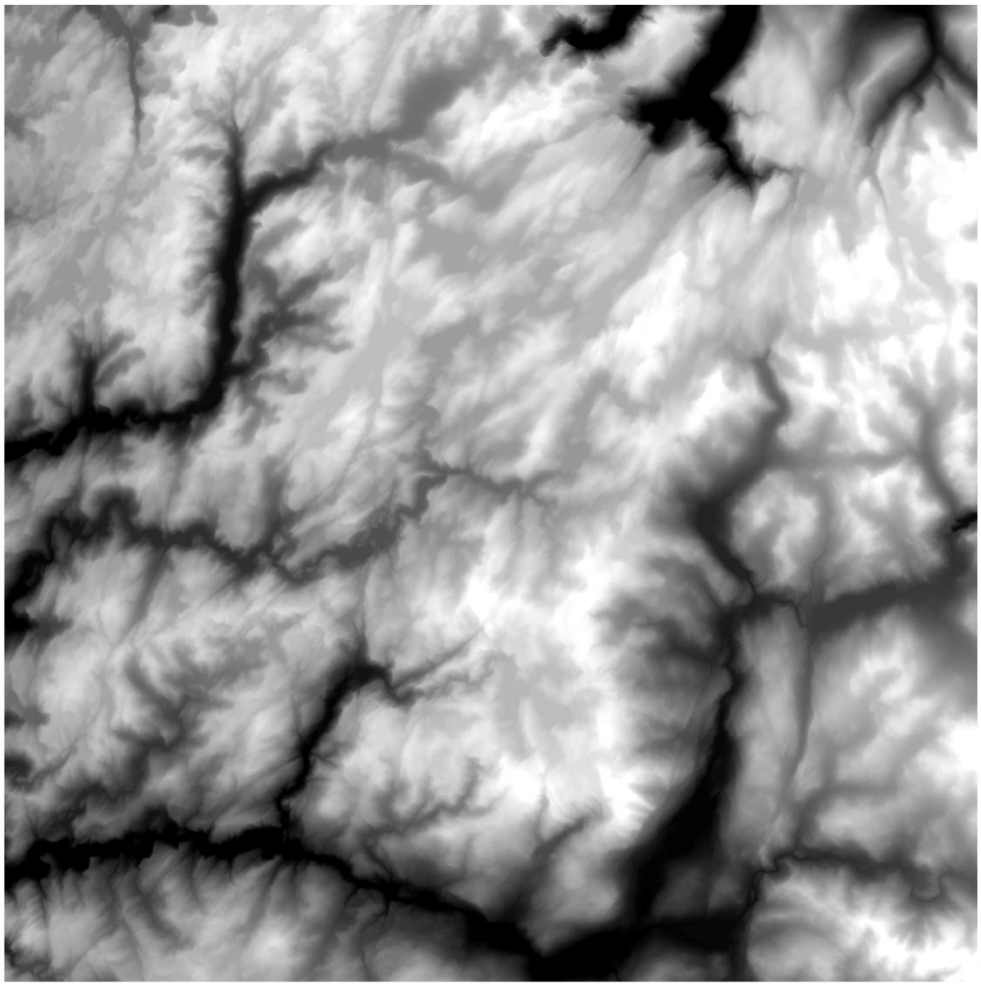


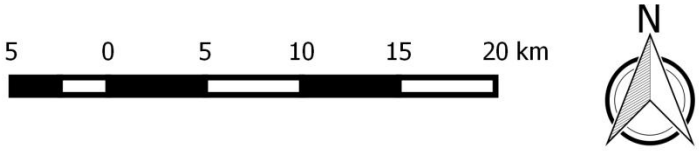
DEM 6700_3_10



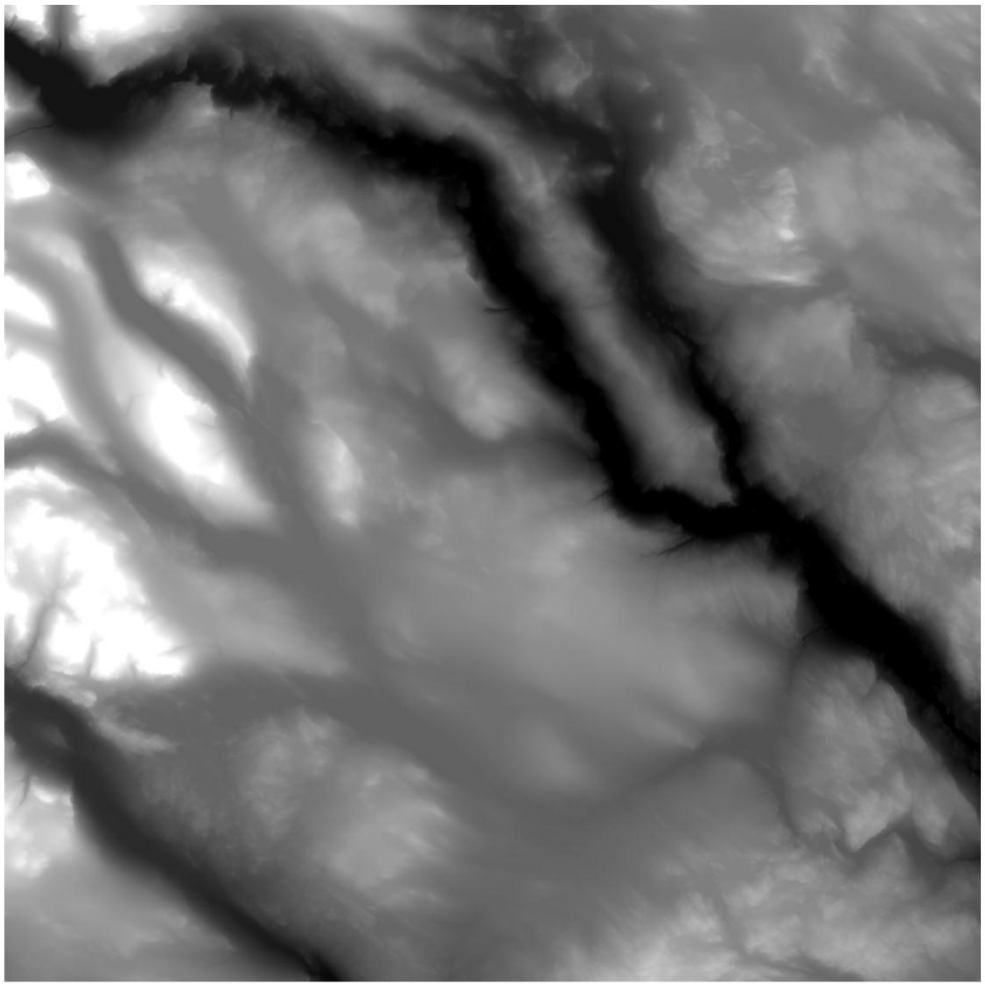


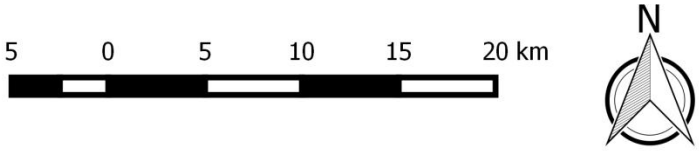
DEM 6700_4_10



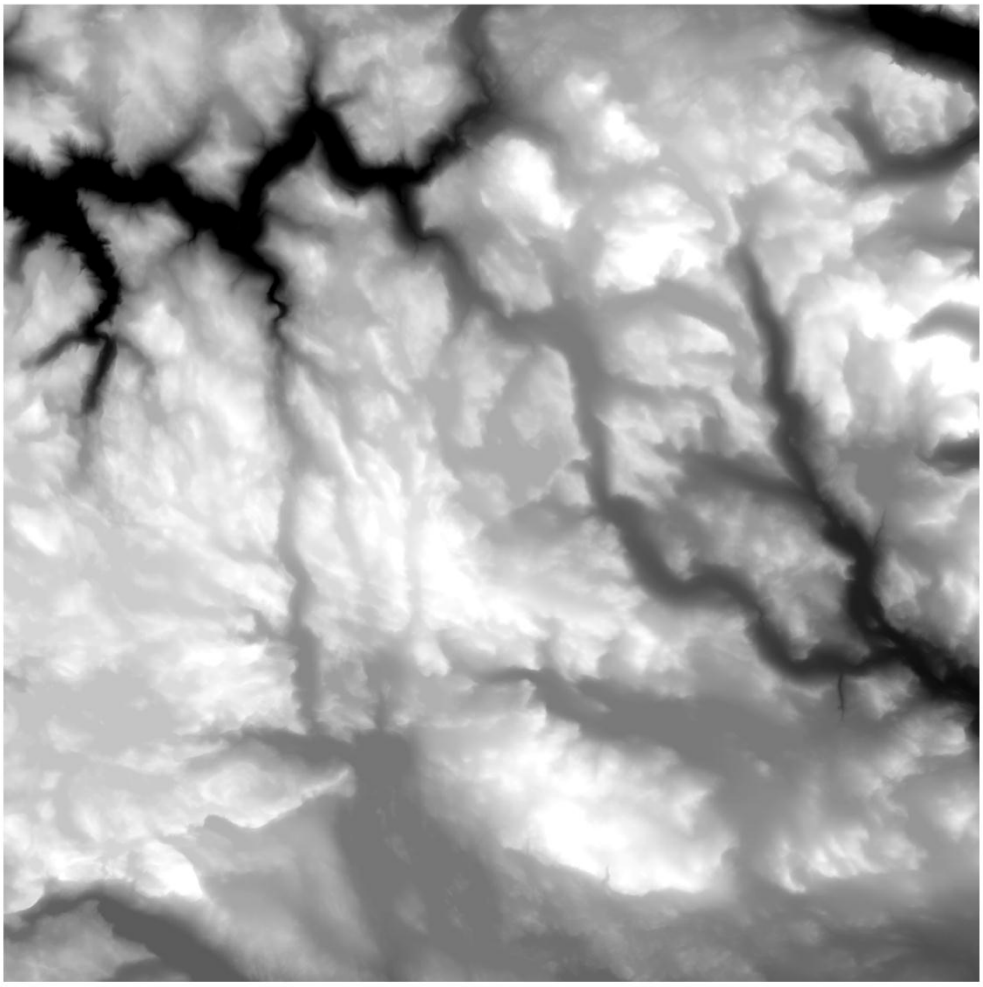


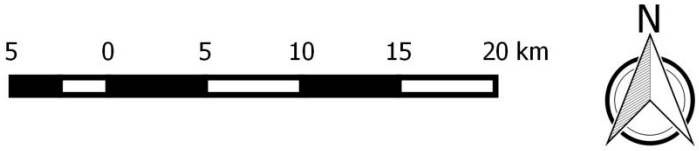
DEM 6701_1_10



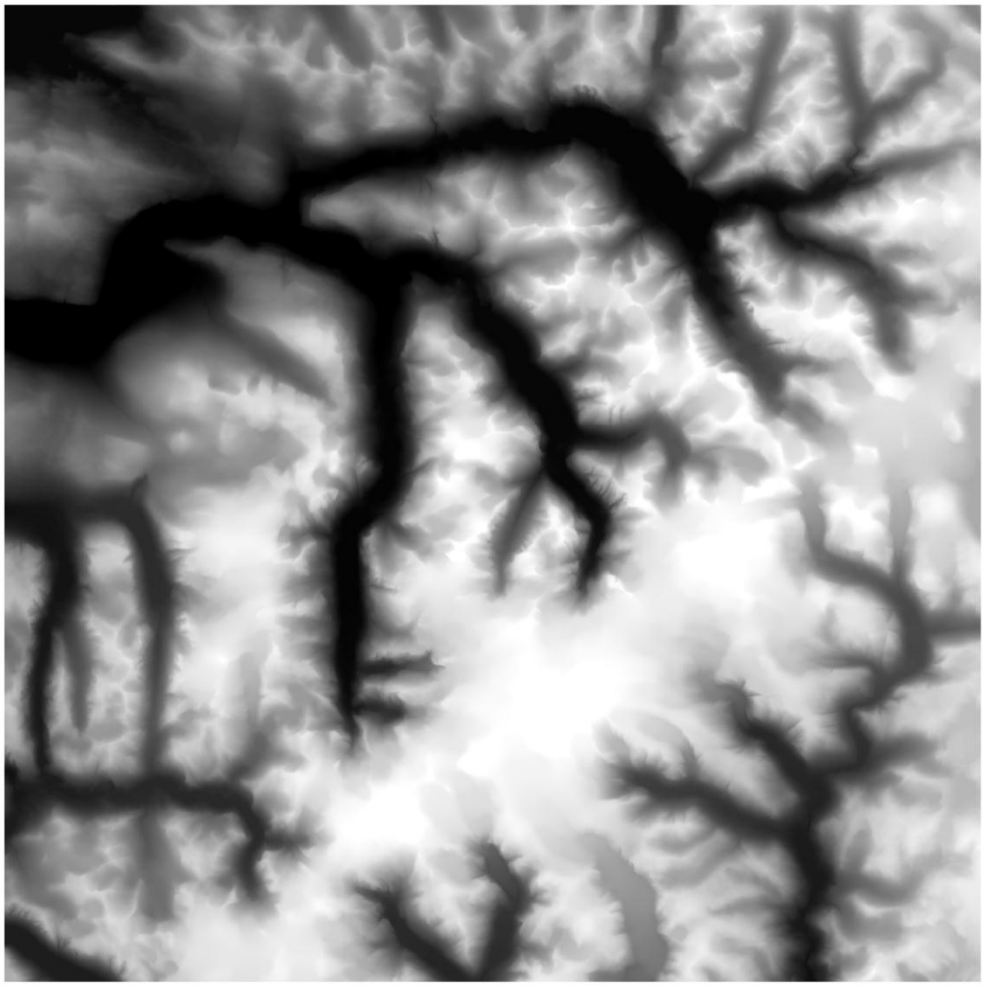


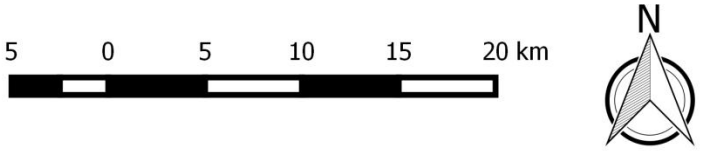
DEM 6701_4_10



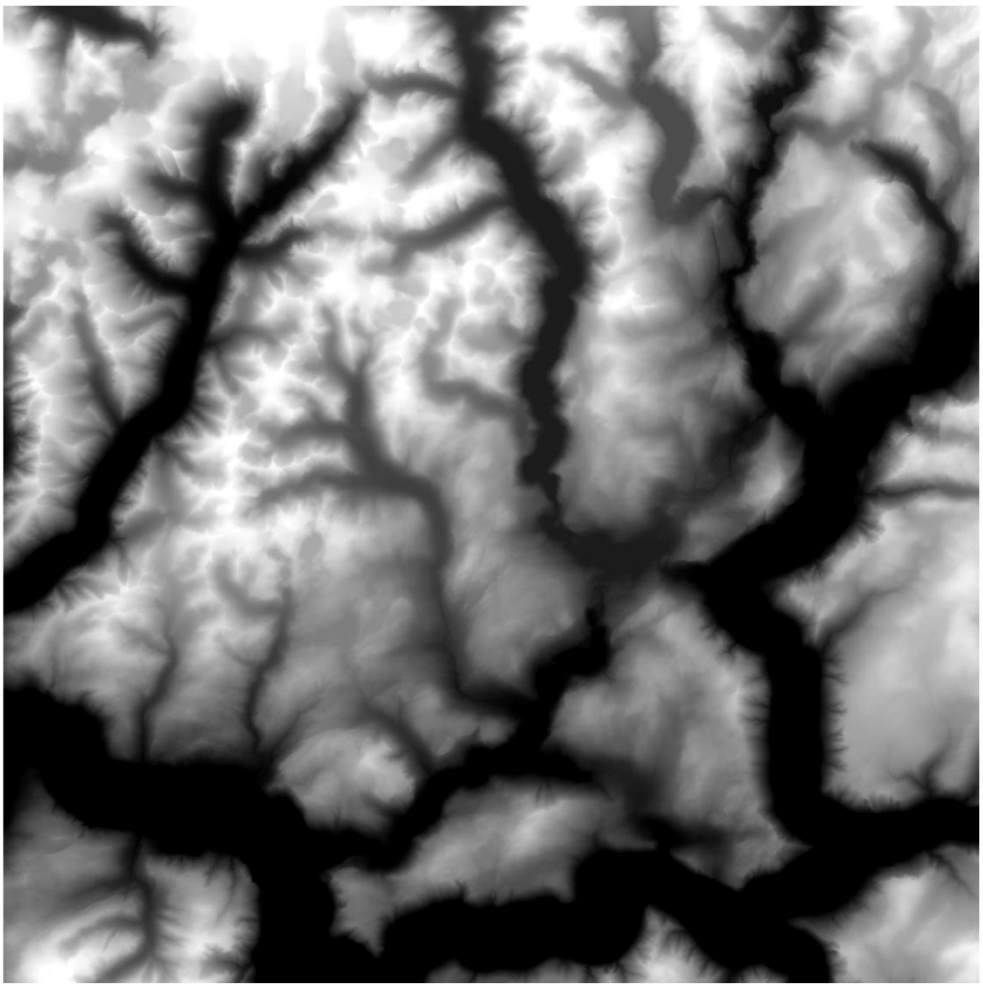


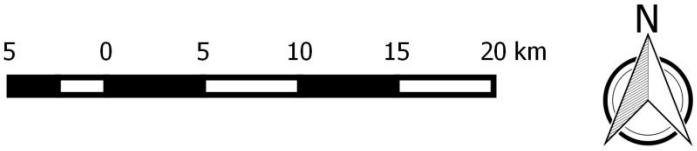
DEM 6800_1_10



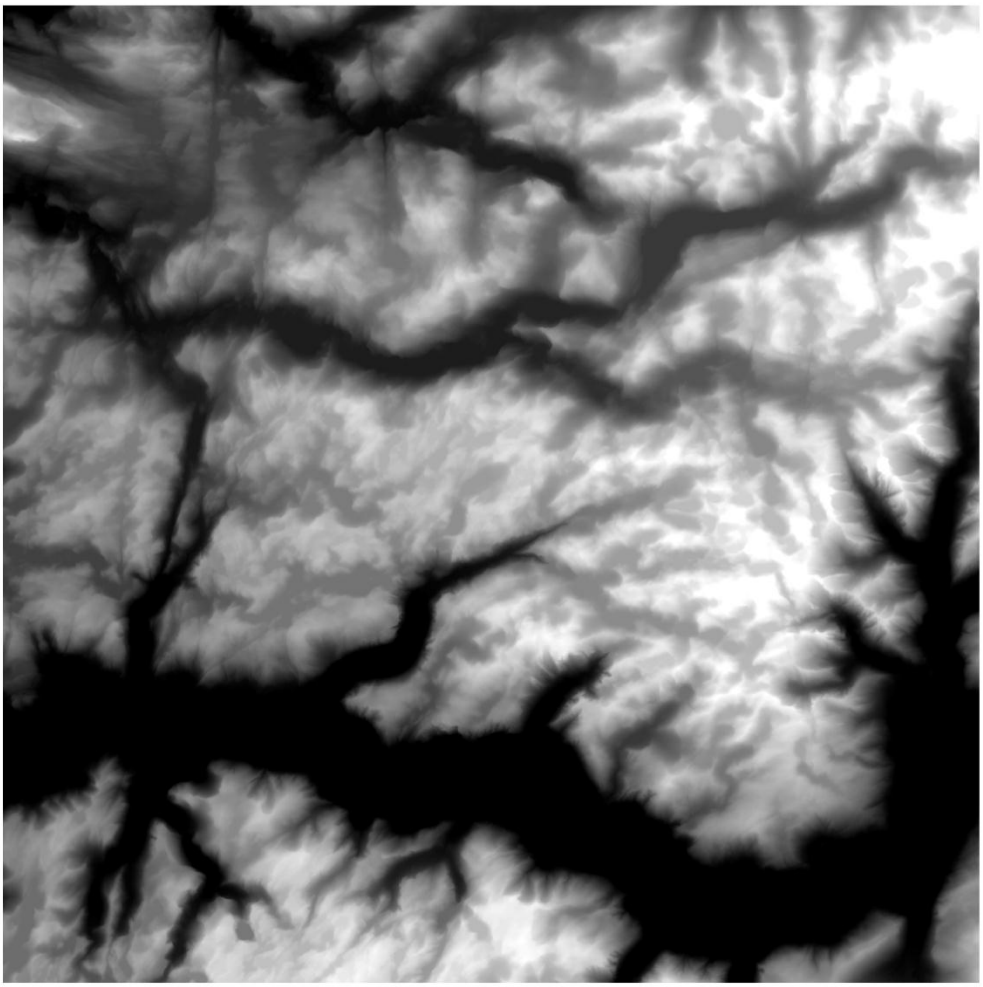


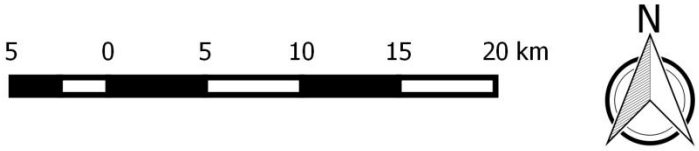
DEM 6800_2_10



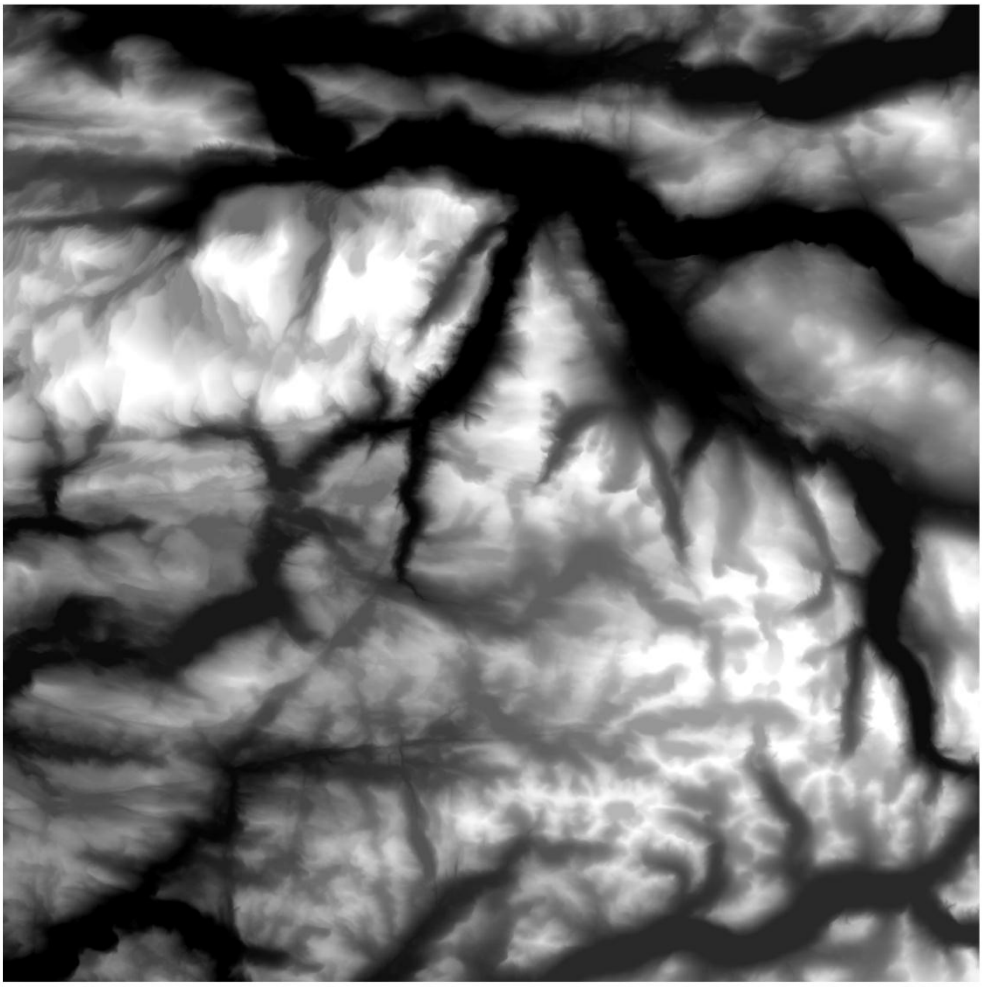


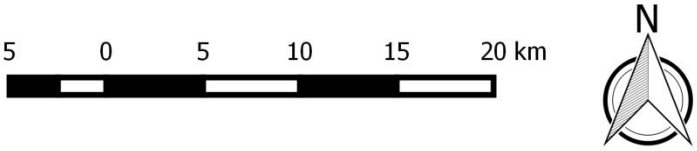
DEM 6800_3_10



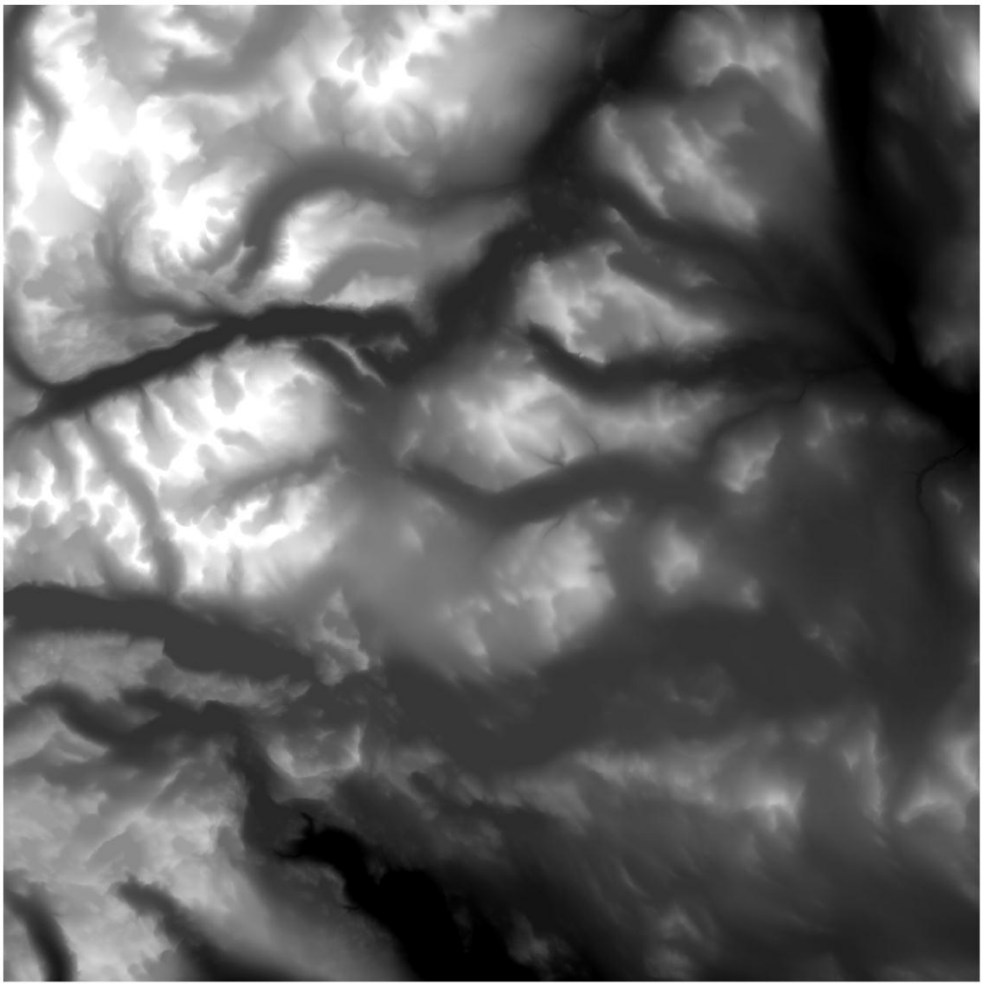


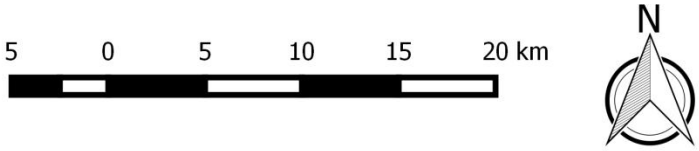
DEM 6800_4_10



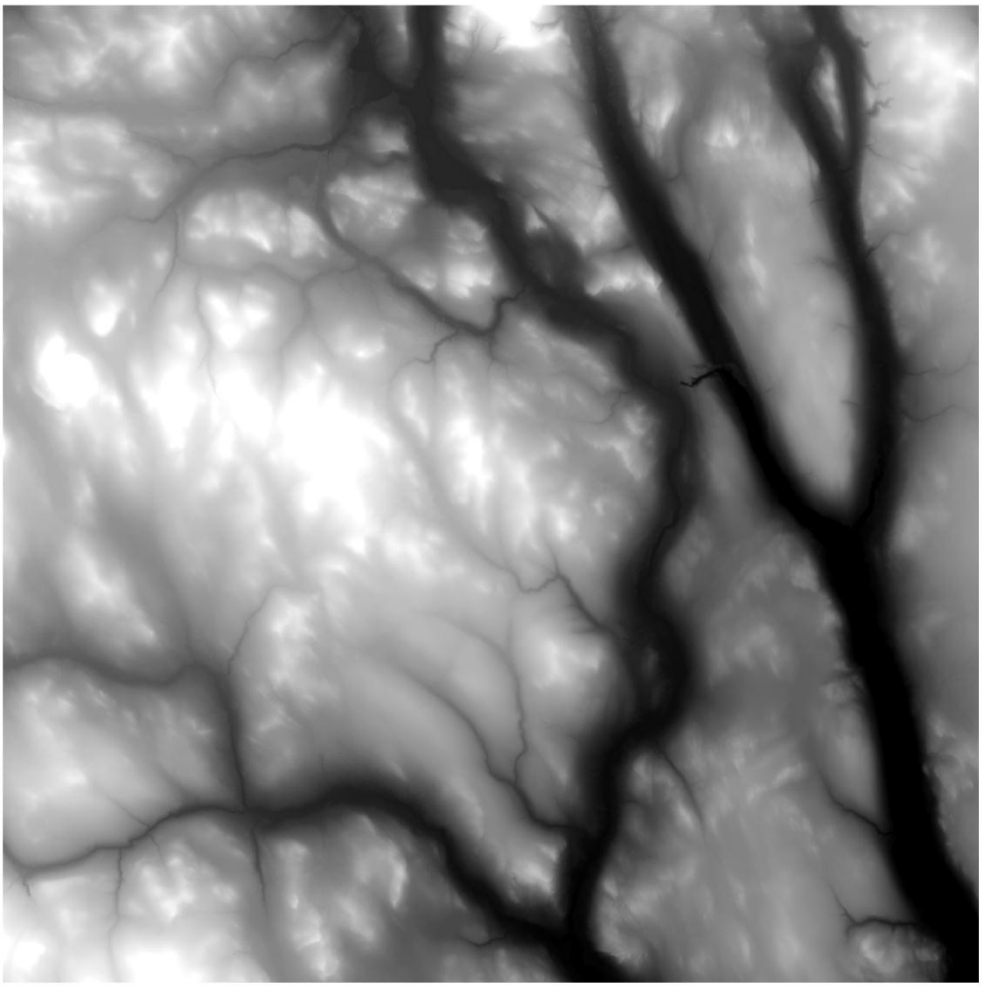


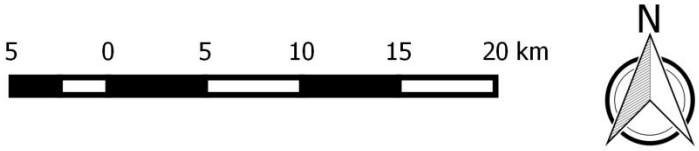
DEM 6801_2_10



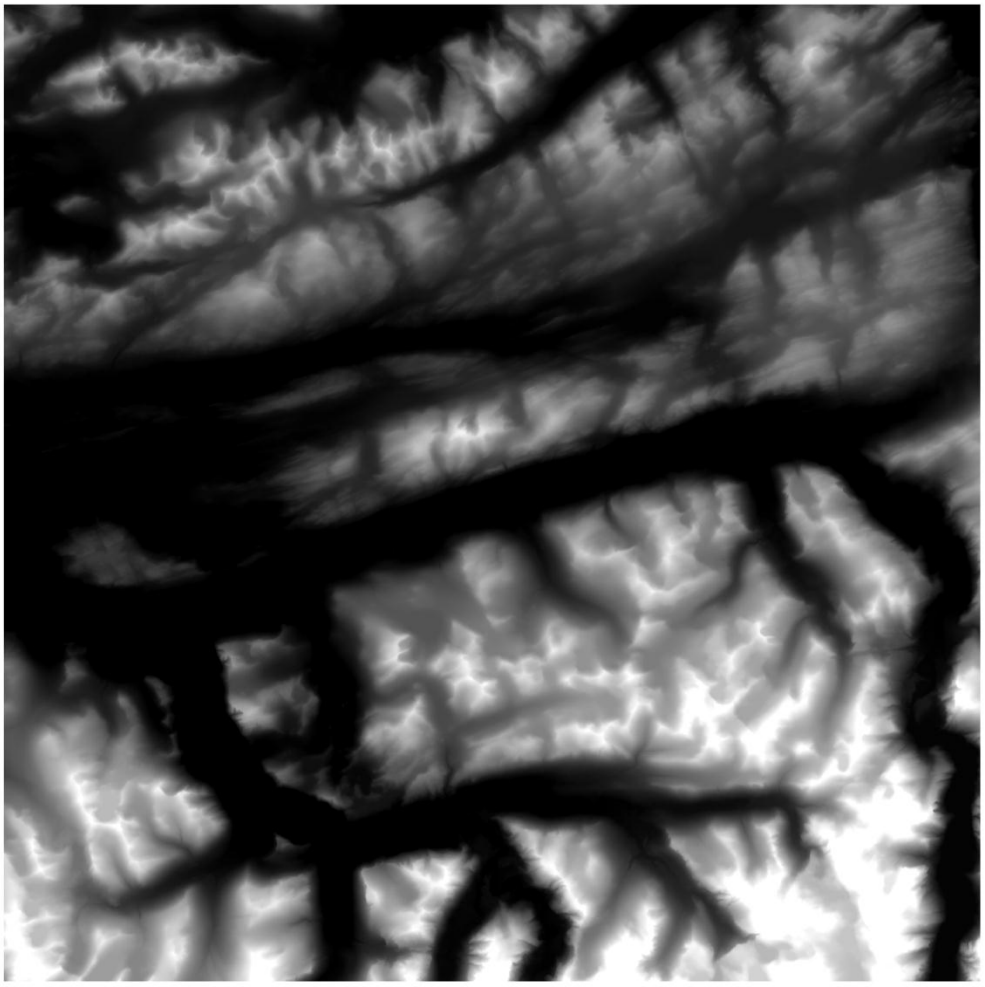


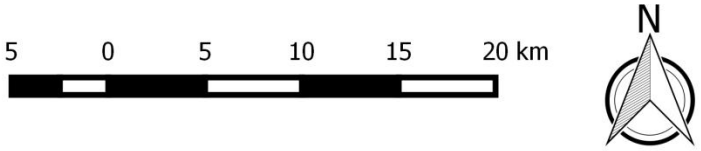
DEM 6802_1_10



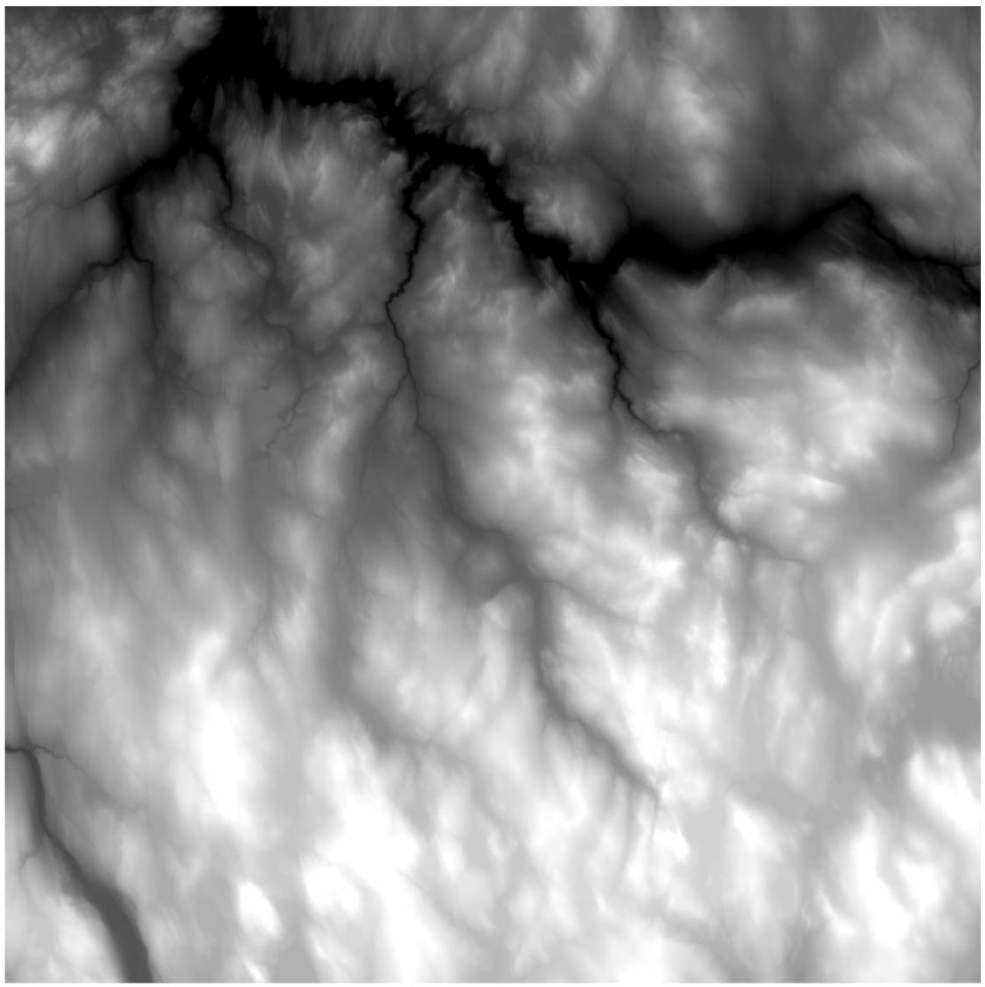


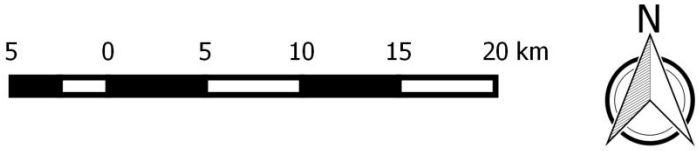
DEM 6901_4_10



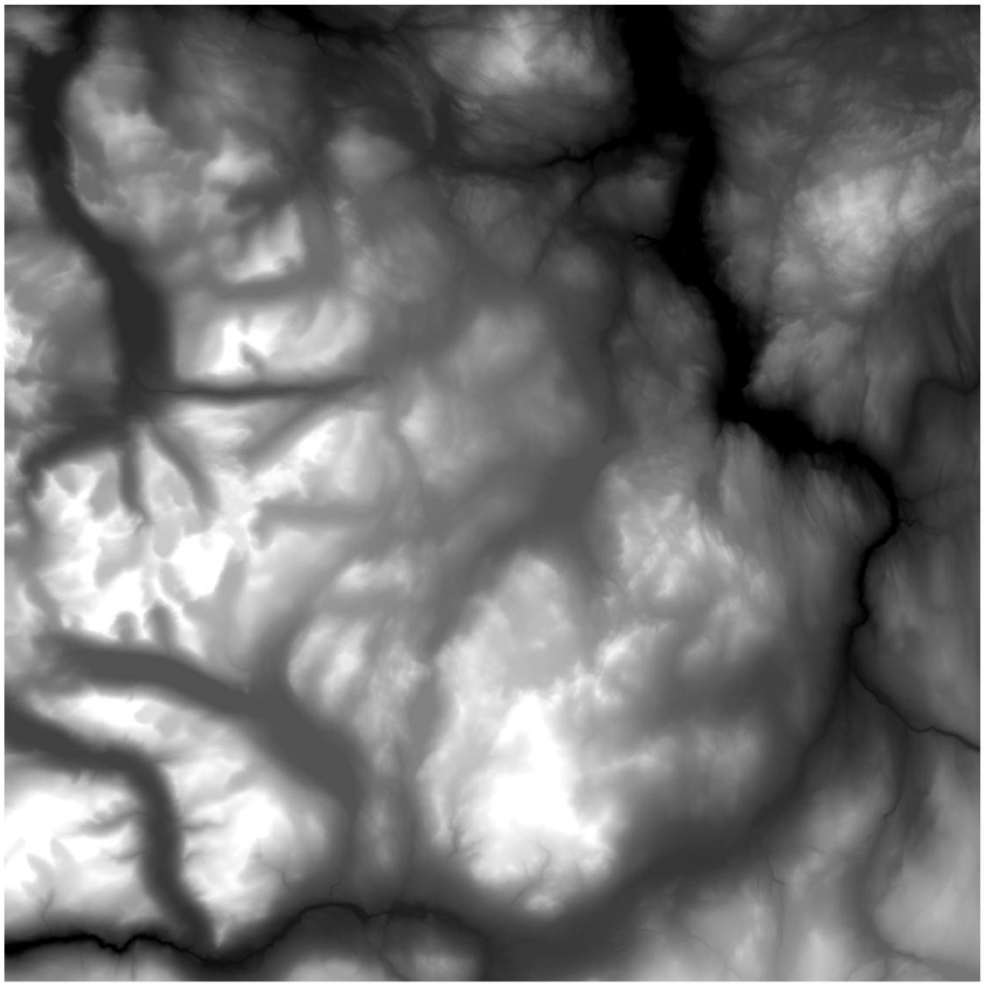


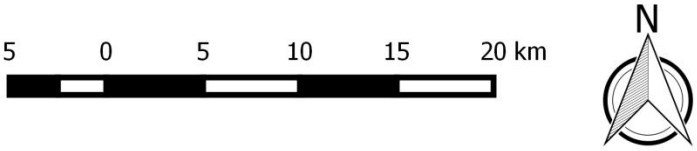
DEM 6902_1_10



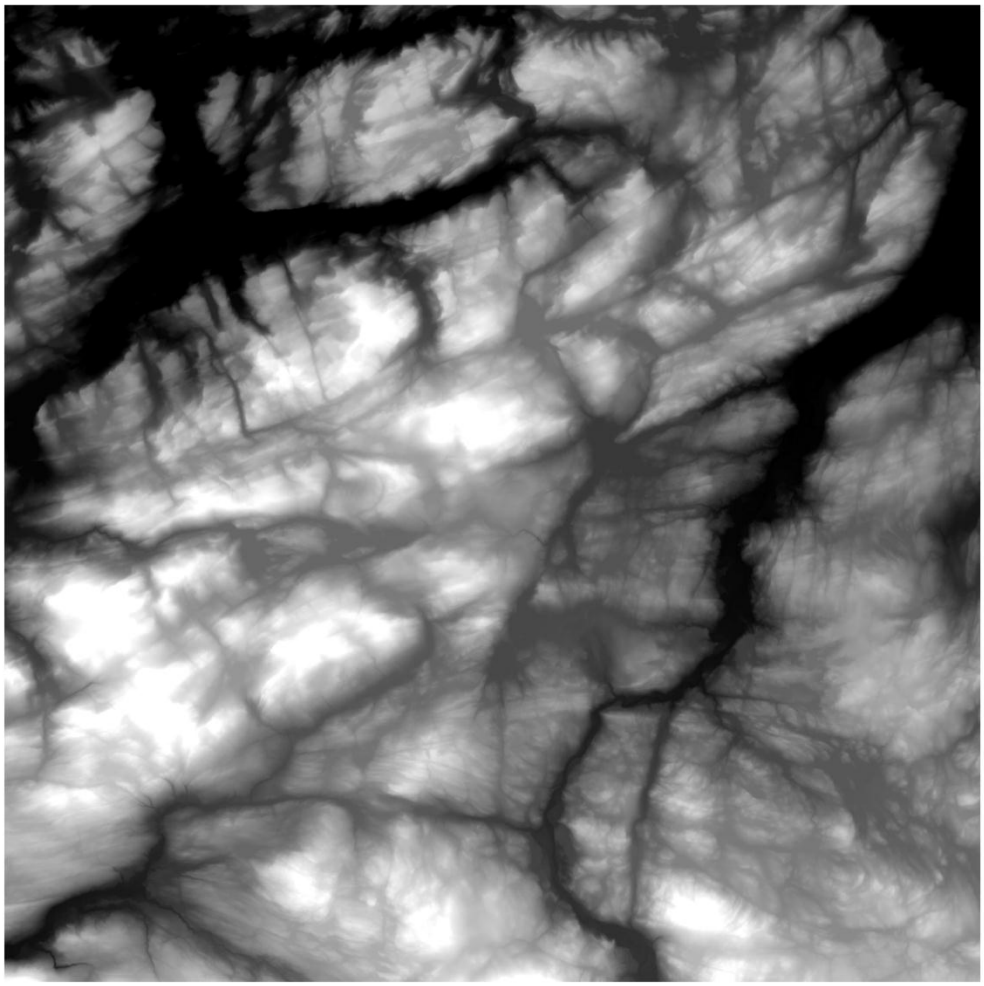


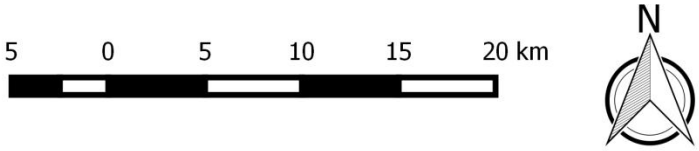
DEM 6902_4_10



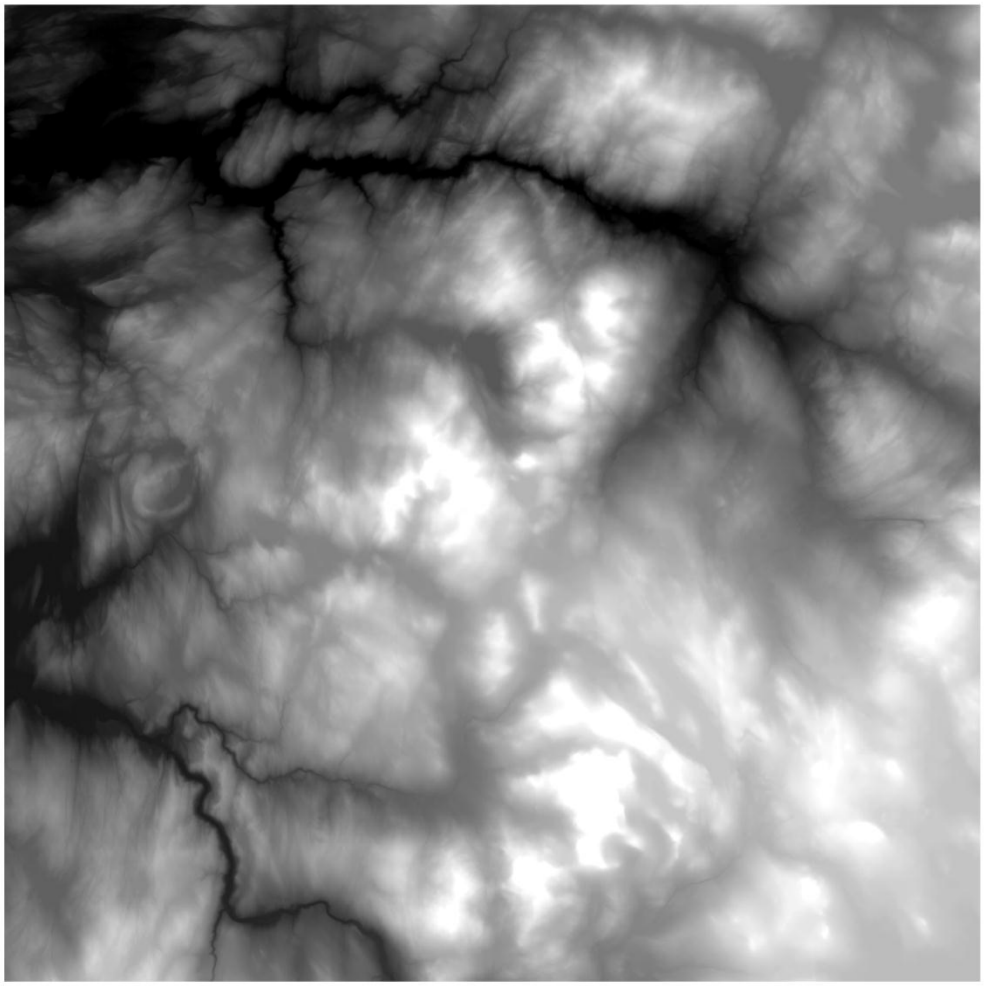


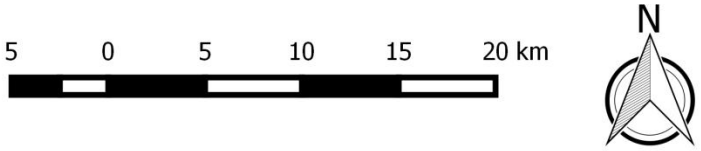
DEM 7002_3_10



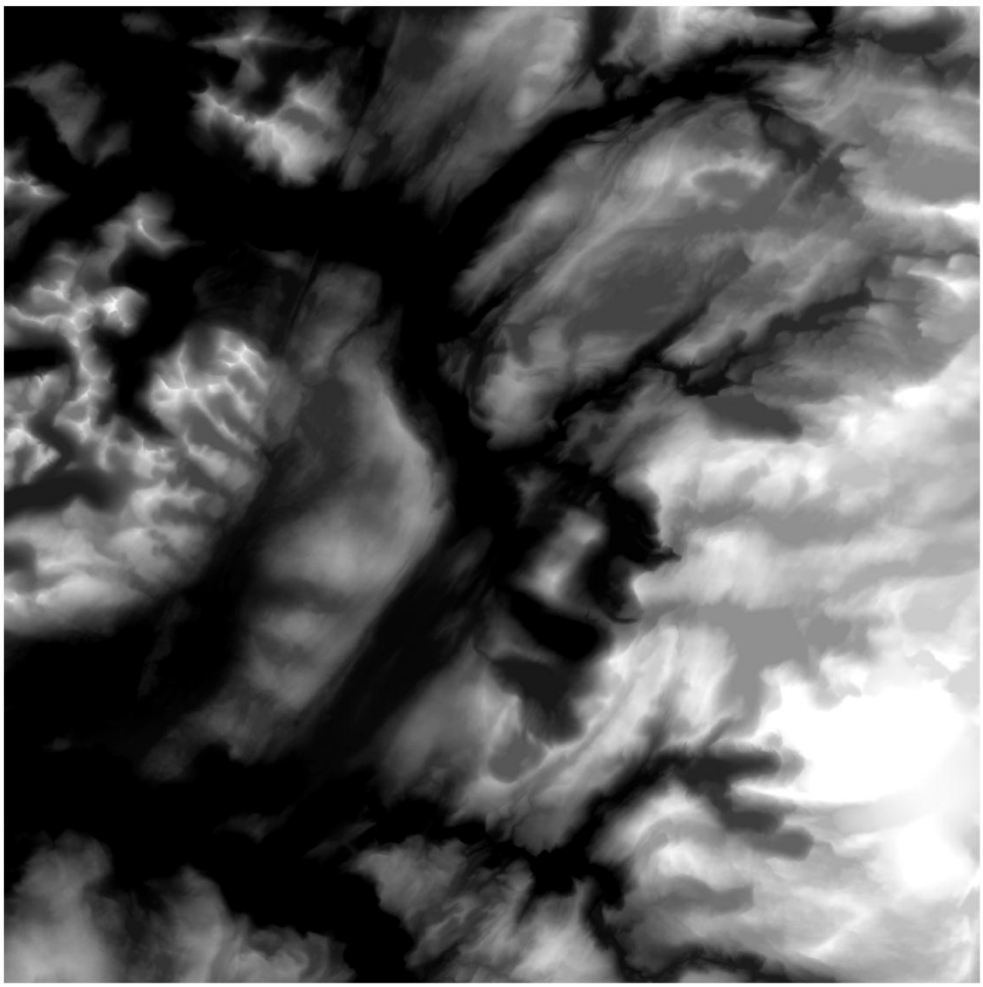


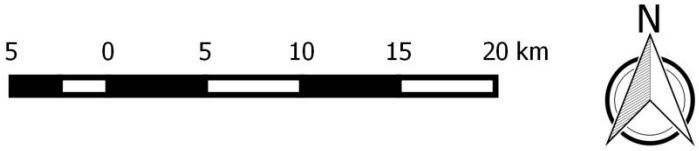
DEM 7003_3_10



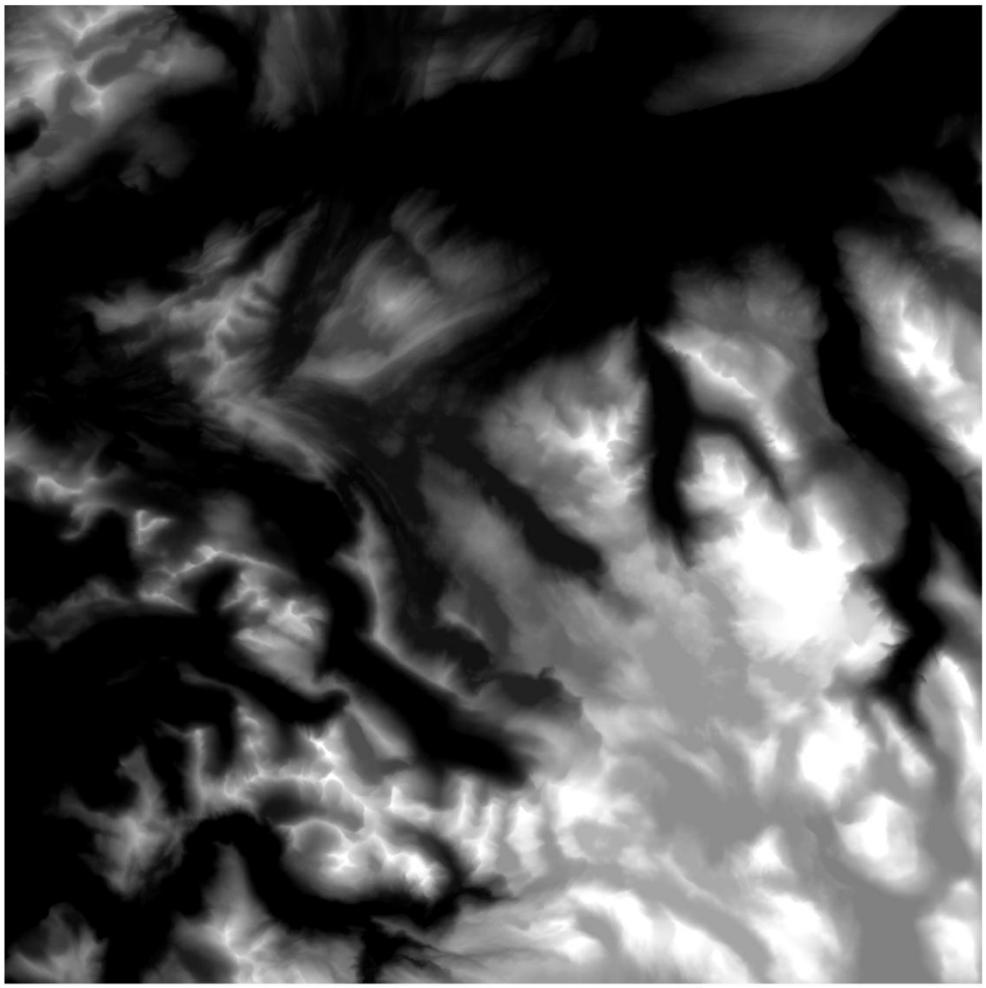


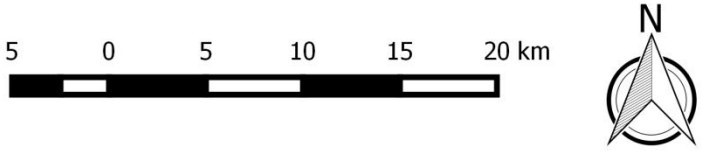
DEM 7405_4_10



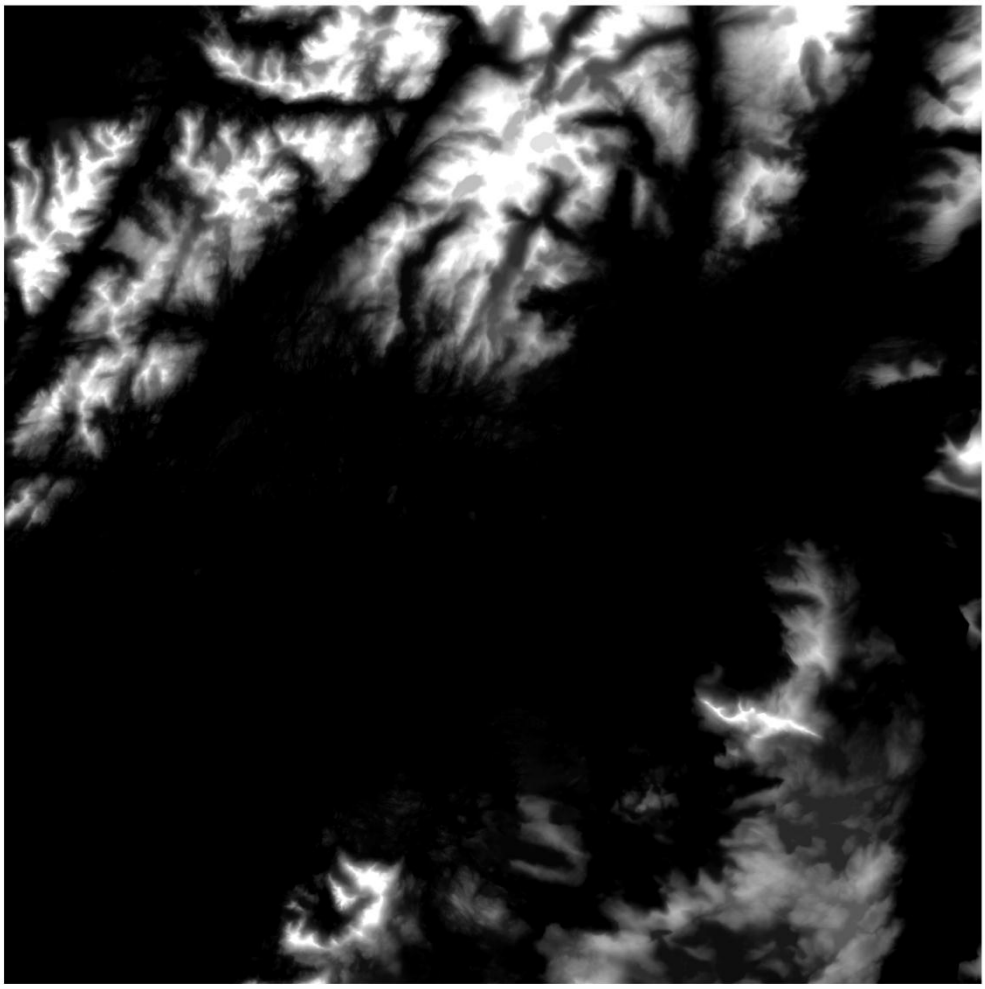


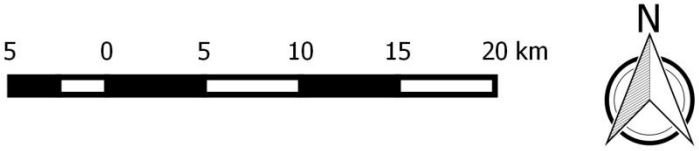
DEM 7505_1_10



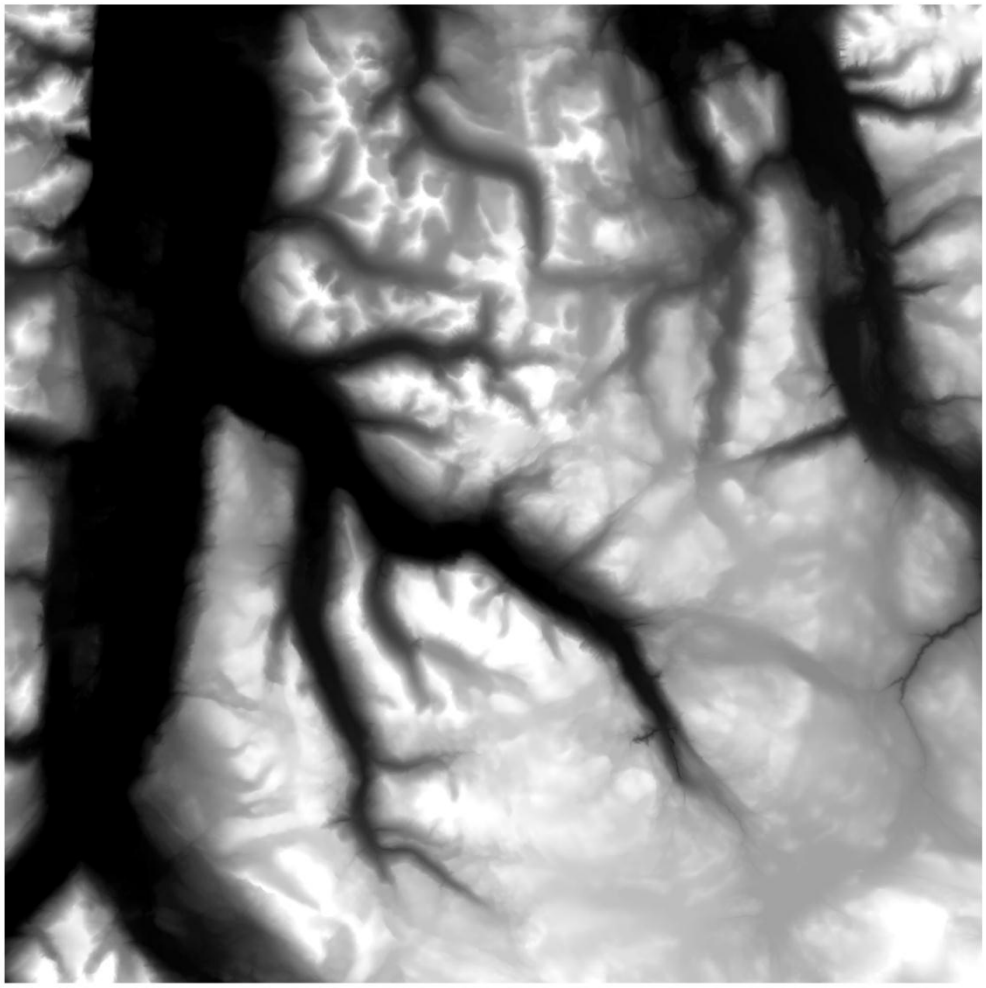


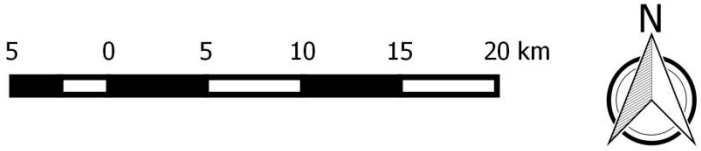
DEM 7505_4_10



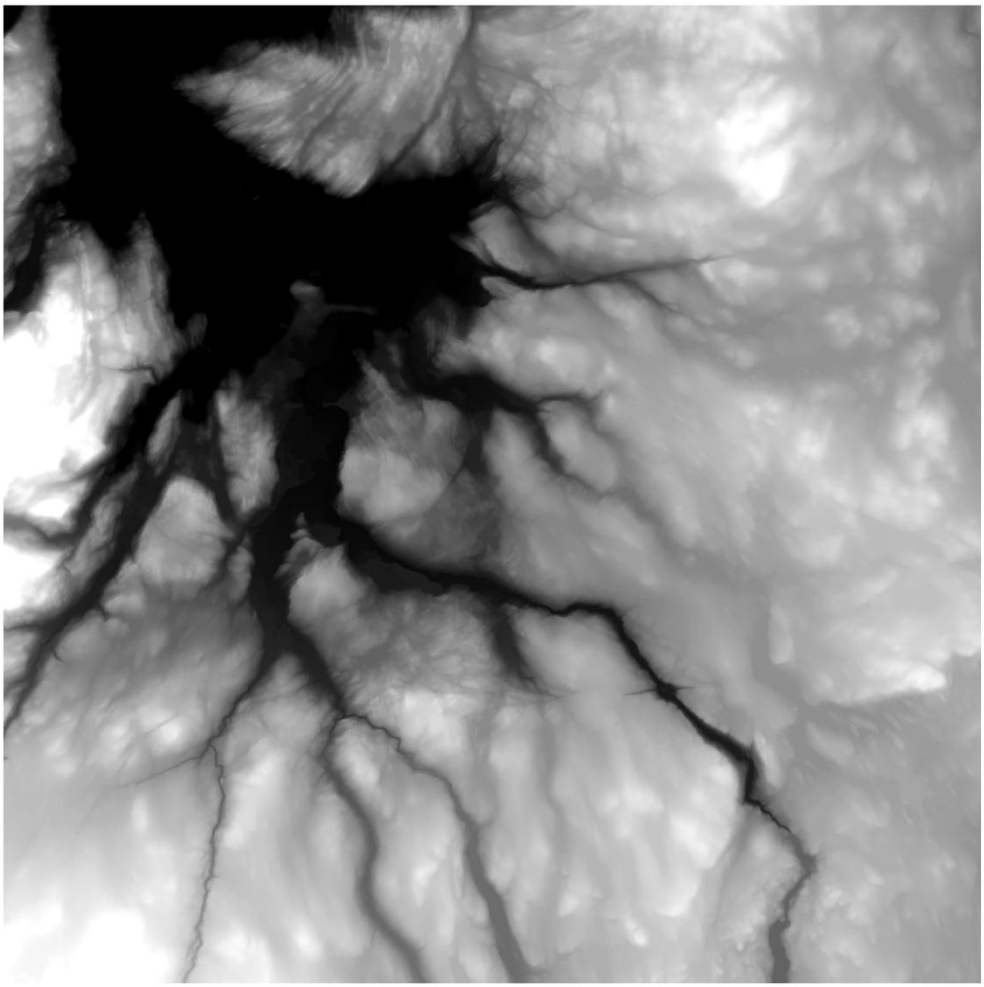


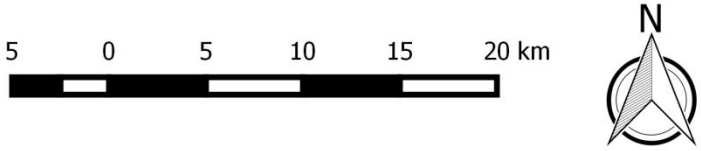
DEM 7707_3_10



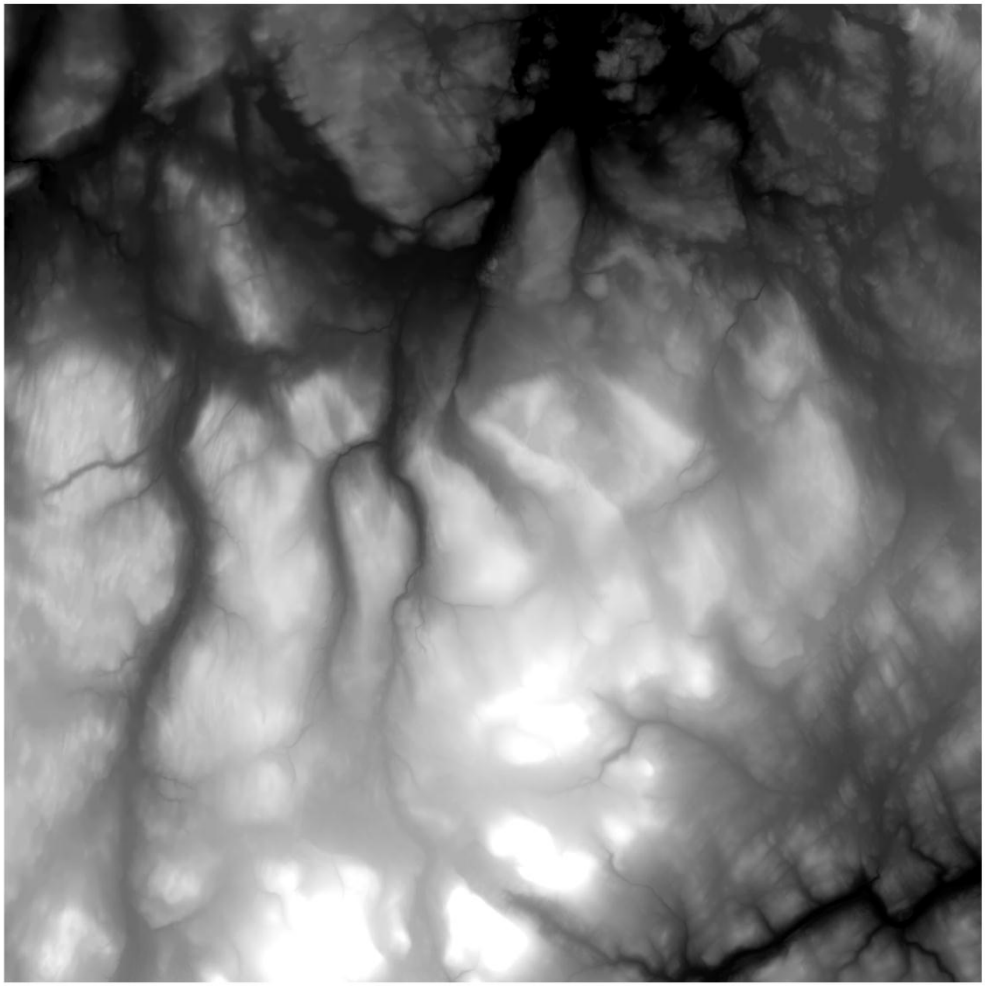


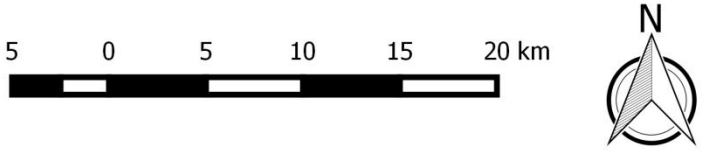
DEM 7708_4_10



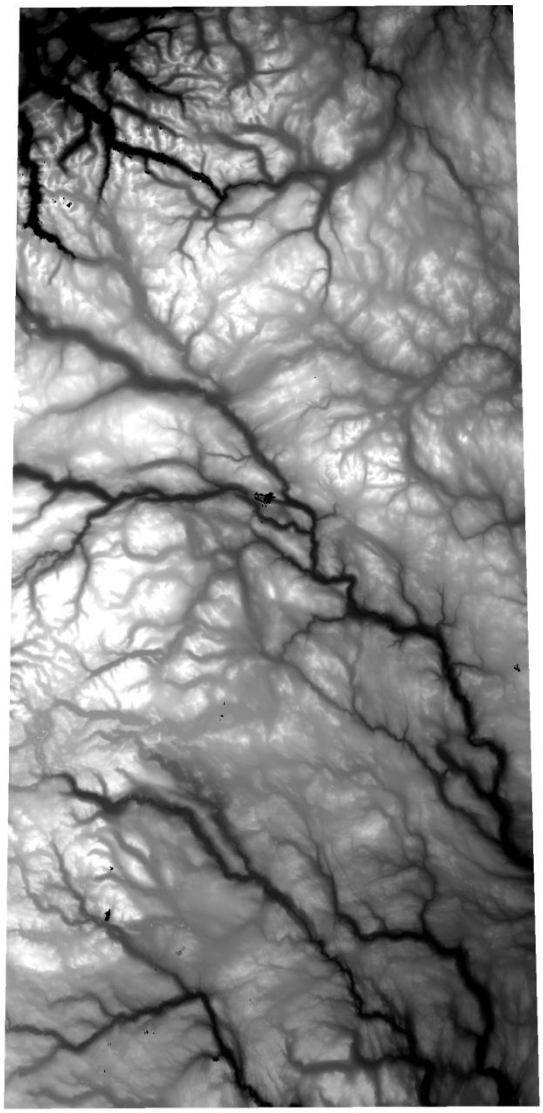


DEM 7809_3_10





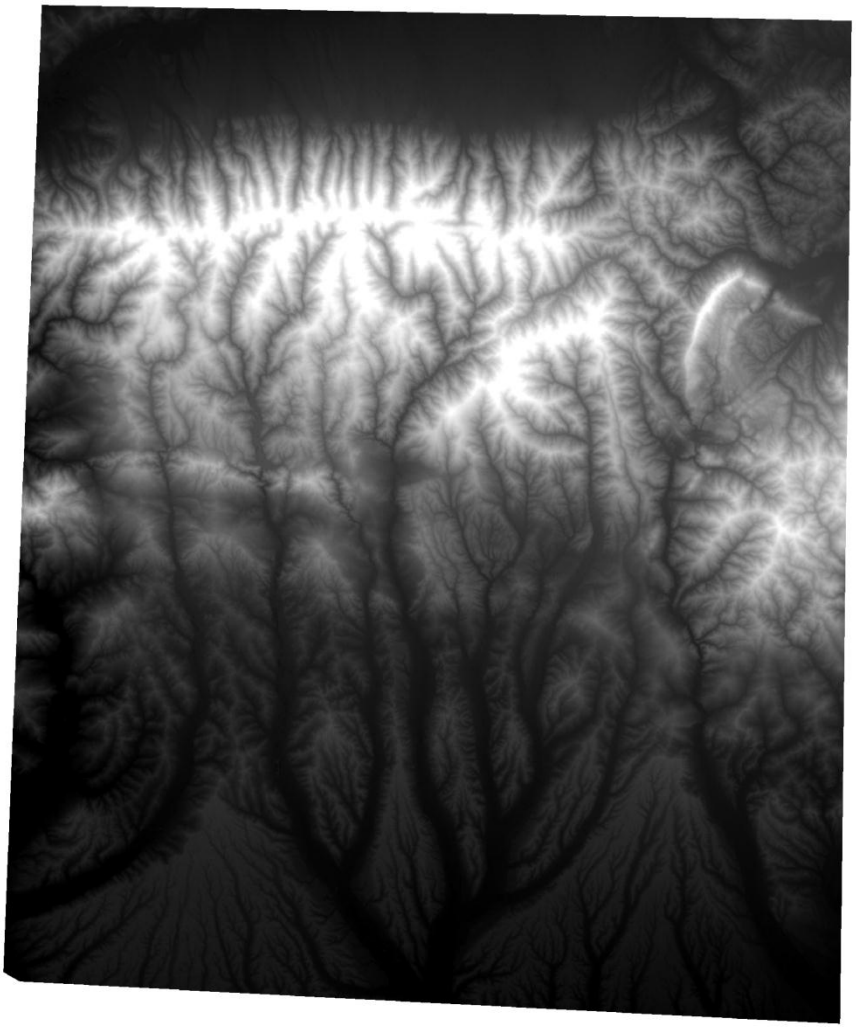
DEM 377911208 Norway



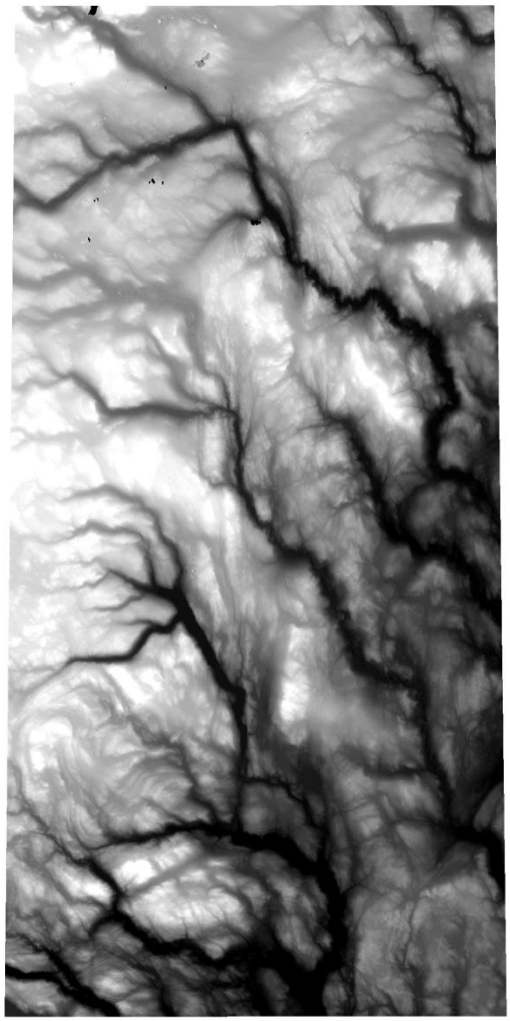
10 0 10 20 30 40 km

N

DEM ASTER Romania



DEM ASTER Norway



DEM SRTM